宝典丛书

100万

MATLAB 宝典

作者拥有10年MATLAB使用经验，对软件理解深刻

全面覆盖数值计算技术，内容深入

采用实例的方式讲解，众多实例都取自工程实践

对Simulink有详细的介绍，适合相关专业人士阅读

陈 杰 编著

電子工業出版社
Publishing House of Electronics Industry
http://www.phei.com.cn

MATLAB 宝典

MATLAB是一个非常庞大的数值计算软件，里面有各种模型库，每个库都需要相应的背景知识。但是万变不离其宗，它有很多基础性的、各个库都需要的技术，这是每个MATLAB用户都需要掌握的。本书的目标就是把所有通用功能一网打尽，让每个MATLAB用户都可以从中汲取到这些知识。

宝 典 丛 书

Excel 2003高级VBA编程宝典
Excel 2003公式与函数应用宝典
Excel应用技巧宝典
中文版Excel 2003宝典
中文版Access 2003宝典
中文版Office 2003宝典
Windows XP宝典（第二版）
Flash 8宝典
Flash 8 ActionScript宝典
Dreamweaver 8宝典
3ds max 7宝典
AutoCAD 2006和AutoCAD LT 2006宝典
Visual Basic .NET编程宝典
Visual C++ 6编程宝典
Struts,Hibernate,Spring集成开发宝典
Java数据库高级编程宝典
Struts数据库项目开发宝典
Hibernate项目开发宝典
JSP宝典
Spring 2.0宝典
Linux宝典（第二版）
Red Hat Linux 9宝典
ASP.NET+SQL Server动态网站设计宝典
Premiere Pro 2宝典
Oracle 10g宝典
Oracle 10g DBA宝典
Protel电路设计与制版宝典
……

ISBN 7-121-03378-X
9 787121 033780 >

责任编辑：张月萍
责任美编：秦 靖

ISBN 7-121-03378-X 定价：89.00元

宝典丛书

MATLAB宝典

陈 杰 编著

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内容简介

本书由浅入深、循序渐进地介绍了MATLAB7.0的知识体系及操作方法。全书共分14章，内容涵盖了MATLAB7.0概述、基础知识、数值运算、数据分析、符号计算、数据的可视化、M语言程序设计、Simulink仿真系统、句柄图形、GUI、文件输入/输出、编译器和应用程序接口等。本书最大的特色在于每一节的例子都是经过精挑细选的，具有很强的针对性，力求让读者通过亲自动手做而掌握基本参数及制作技巧，学习尽可能多的知识。

本书适用于初、中级MATLAB7.0用户，同时也可作为本科生、研究生和教师以及广大科研工作人员学习MATLAB的参考用书。

图书在版编目(CIP)数据

MATLAB宝典/陈杰编著．—北京：电子工业出版社，2007.1
（宝典丛书）
ISBN 7-121-03378-X

Ⅰ.M… Ⅱ.陈… Ⅲ.计算机辅助计算－软件包，MATLAB 7.0 Ⅳ.TP391 75

中国版本图书馆CIP数据核字（2006）第129641号

责任编辑：张月萍　　特约编辑：明足群
印　　刷：北京天竺颖华印刷厂
装　　订：三河金马印装有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　邮编：100036
开　　本　787×1092　1/16　印张　57　字数：1623千字
印　　次：2007年1月第1次印刷
定　　价：89.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系电话：（010）68279077，邮购电话：（010）88254888。

质量投诉请发邮件至zlts@phei.com.cn。

服务热线：（010）88258888。

前　言

MATLAB是Mathworks公司推出的一套高性能数值计算和可视化软件，它集数值分析、矩阵运算、信号处理和图形显示于一体，在系统建模和仿真、科学和工程绘图以及应用程序开发等方面有着广泛应用。MATLAB由著名的线性代数软件包LINPAK和特征值计算软件包EISPACK的子程序为基础，发展为一种开发性程序设计软件，因此MATLAB已经由简单的矩阵计算分析软件发展成为通用性极高、带有多种实用工具的运算操作平台。

MATLAB7.0是Mathworks公司近年推出的最新版本，相对于以前的版本，MATLAB7.0有了较大的改进和增补，在开发环境、程序设计、数值处理以及数据可视化方面提供了许多新功能和更为有效的处理方法。

为了帮助众多从业者提高软件使用及操作水平，笔者精心编著了本书。本书依照读者的学习规律，首先介绍基本概念和基本操作，在读者掌握了这些基本概念和基本操作的基础上，再对内容进行深入的讲解，严格遵循由浅入深、循序渐进的原则。本书按照MATLAB7.0内在的联系将各种工具、命令和命令面板交织编排在一起，这样编排虽然不像帮助文档那样有层次感，但是对理解和掌握MATLAB7.0却是大有帮助的。

本书在内容的编排和目录组织上都十分讲究，争取让读者能够快速掌握软件的使用方法。讲解具体知识的时候，尽量避免冗长的知识讲解，直接切入主题，告诉读者如何实现特定功能，让读者在实际操作中熟悉软件的使用。严格制作每一个实例，强调实例效果，这样保证读者的起步层次比较高，在实践的过程中提高制作水平。

和其他书籍相比，本书有以下特点。

1. 内容全面，权威

本书内容全面，它不但表现在介绍某一专项技术时做到全方位的渗透，而且即使本书介绍的仅是MATLAB的某项应用，也会做到对各个技术点逐一攻破。“权威”是指全书的介绍准确，同时还有一定的高度。

2. 取材广泛，内容充实

作者在讲解每一个知识点之前，充分考虑了MATLAB的知识和实践工作的结合，精心挑选数学研究、图形设计、工程运用等各个领域的应用，使读者不仅仅学到MATLAB的操作技巧，而且对创意、思路有所提高。

3. 内容深入

本书的所有实例都有一定的代表性和通用性，并不是为单纯介绍某个命令而选取的，因此有些实例的步骤比较多，综合了MATLAB中的多个知识点，能够提高用户综合使用知识的能力。

4. 讲解仔细

每个实例的制作步骤都以通俗易懂的语言阐述，并穿插讲解和技巧文字，在阅读时就像听课一样详细而贴切。读者只需要按照步骤操作，就可以学习到MATLAB的相关功能。

本书包括的内容

第1章首先带领读者了解MATLAB7.0的应用领域，了解MATLAB7.0的新增功能，熟悉MATLAB7.0的用户界面等入门知识。

第2章介绍MATLAB7.0的基础知识。包括在MATLAB中如何创建数组、操作数组、操作矩阵等基础知识。

第3章主要介绍MATLAB7.0的数值运算，主要包括矩阵分析、线性方程组、矩阵分解、数值积分、函数零点和数理分析等内容。

第4章主要介绍如何使用MATLAB7.0进行数据分析，主要内容包括数据插值、曲线拟合、傅里叶分析、优化和常微分方程等。

第5章主要介绍如何使用MATLAB7.0进行符号计算，主要内容包括符号表达式、符号表达式的运算、符号函数的操作、符号微积分、符号积分变换、符号矩阵计算、符号线性代数方程和符号微分方程等。

第6章主要介绍如何使用MATLAB7.0进行数据可视化，主要内容包括如何绘制二维图形、三维图形以及如何操作三维图形等。

第7章主要介绍如何使用MATLAB7.0进行程序设计，主要内容包括程序设计结构、控制语句、程序的向量化以及如何调试分析程序代码等。

第8章主要介绍Simulink仿真系统的知识，主要内容包括Simulink基础知识、数据类型、基础操作和仿真设置等内容。

第9章主要介绍Simulink仿真系统的高级技术，详细介绍了子系统、封装子系统、使能子系统、触发子系统和S-函数等内容。

第10章主要介绍句柄图形的知识，主要内容包括句柄图形体系、句柄对象操作和高层绘图命令等内容。

第11章主要介绍图形用户界面的制作，主要内容包括如何使用M文件创建GUI、如何使用GUIDE创建GUI、如何添加菜单对象、如何添加控件等。

第12章主要介绍文件的输入和输出，主要内容包括如何在MATLAB中处理外部的二进制文件、文本文件和图像等。

第13章主要介绍MATLAB7.0中的编译器，主要内容包括编译器的安装、设置和对应的命令，以及如何创建独立应用程序等。

第14章主要介绍应用程序接口，主要内容包括如何创建MEX文件、MAT文件，以及如何使用MATLAB7.0的引擎技术等。

本书具有知识全面、实例精彩、指导性强的特点，力求以全面的知识性及丰富的实例来指导读者逐渐学习MATLAB7.0各方面的技术。本书适用于初、中级MATLAB7.0用户，同时也适合使用MATLAB的本科生、研究生和教师以及广大科研工作人员作为参考用书，对高级读者也有一定的启发意义。

本书中的MATLAB文件请读者访问http://www.hxex.cn的“资源下载”栏目查找并下载。

作　者

2006年12月

目　　录

第1章 MATLAB概述

本章包括

- MATLAB7.0的安装
- MATLAB7.0的工作环境
- MATLAB7.0的常见命令
- MATLAB7.0的帮助系统

[illegible]

1.1 MATLAB7.0简介

[illegible]

- **开发环境**：[illegible]
- **代码开发**：[illegible]
- **数值处理**：[illegible]
- **数据可视化**：[illegible]
- **文件I/O和外部应用程序接口**：[illegible]

1.2 MATLAB7.0的安装

[illegible] MATLAB7.0在PC机的Windows操作系统中的典型安装方法。

[illegible]

[illegible]

[illegible]

[illegible] 大改变。下面主要介绍几个明显改变的界面。

[illegible]，如图 1.1 所示。

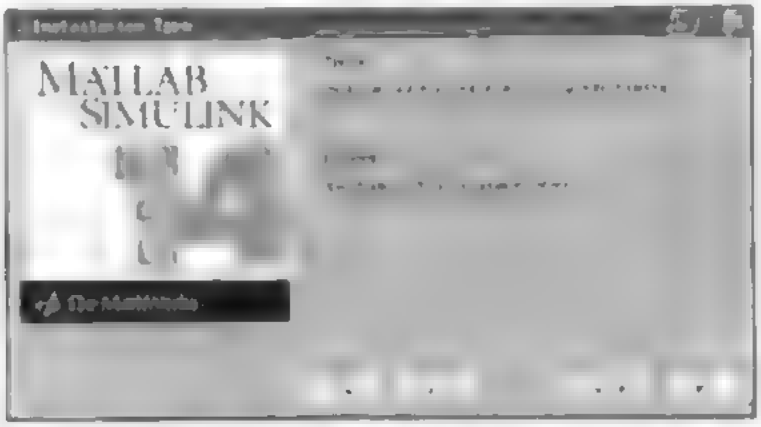


图 1.1　选择自定义安装选项

[illegible]，单击界面中的 "Next" 按钮，进入 MATLAB 安装的下一个界面，用户可以在该界面中选择需要安装的组件，如图 1.2 所示。

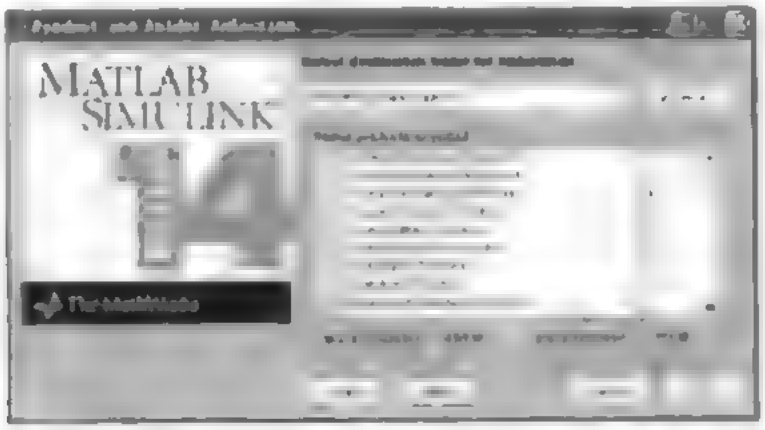


图 1.2　MATLAB 组件选择界面

[illegible]

[illegible] MATLAB 安装目录的大小，[illegible]，[illegible] MATLAB 程序，[illegible] MA、AB 也在其[illegible]

[illegible]

[illegible]

表 1.1 MATLAB的安装组件

组件名称	说明
[illegible]	[illegible]整个软件的核心部分，为整个软件系统提供MATLAB工作环境
[illegible]	[illegible]
Symbolic Math	些组件
Optimization	
Matlab Compiler	
Control System	MATLAB的更加专业性组件，建议用户根据需要选择组件
Curve Fitting	
Statistics	
[illegible]	

[illegible]

[illegible]界面，用户可以在界面中选择MATLAB的使用选项，如图1.3所示。

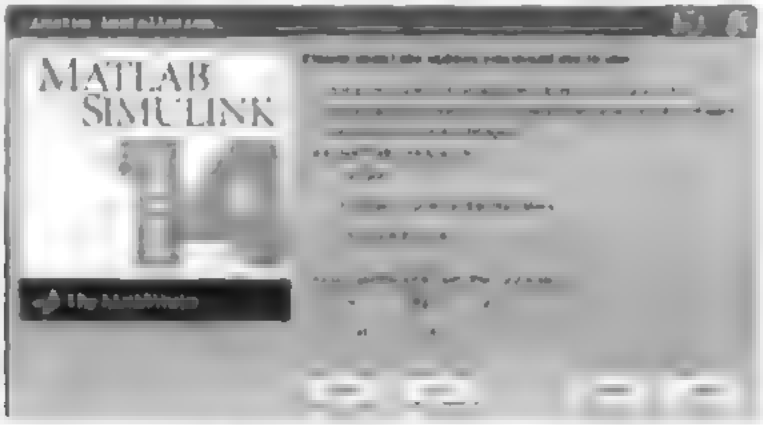


图 1.3 MATLAB选项界面

[illegible]

[illegible]选项。这样就可以很方便地启动MATLAB。

[illegible]，在默认的情况下，系统使用MATLAB打开这些扩展名的文件。

1.3 MATLAB7.0的工作环境

[illegible]，这将MATLAB[illegible]在MATLAB7.0的工作环境。

[illegible]

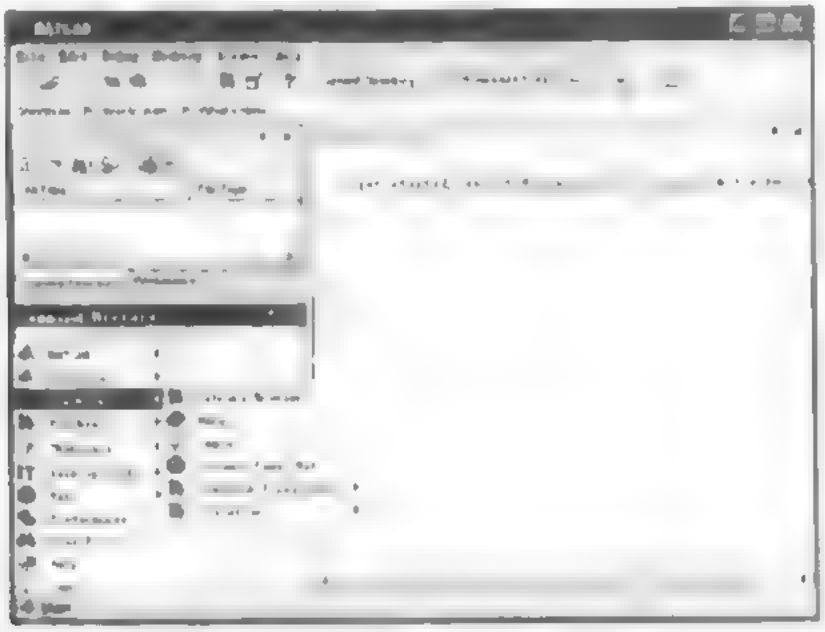


图1.4 MATLAB操作界面的默认外观

[illegible]，在为MATLAB的操作界面环境[illegible]，[illegible]目录，[illegible]MATLAB[illegible]

[illegible]

1.3.1 操作界面简介

MATLAB[illegible]的默认操作界面。

[illegible]，MATLAB[illegible]面。同时，在窗口的左下角为“开始”按钮。

[illegible]

下面详细介绍MATLAB中常见的几个交互界面。

- **命令窗口（Command Window）**：[illegible]
- **历史命令窗口（Command History）**：[illegible]
- **当前目录窗口（Current Directory）**：[illegible]等。同时，用户可以编辑M文件等。
- **工作空间浏览器（Workspace Browser）**：[illegible]小。用户可以在这个窗口中观察、编辑和提取变量。
- **开始按钮（Start）**：[illegible]

1.3.2 运行命令窗口（Command Window）

[illegible]操作命令和结果都需要在命令窗口中进行。

[illegible]使命令窗口版本操作界面。得到的命令窗口如图1.5所示。

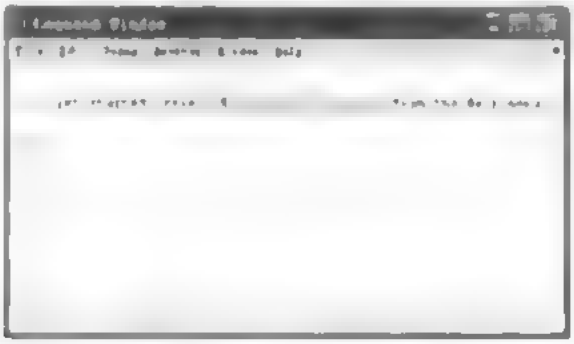


图1.5 单独的命令窗口

[illegible]

示符“>>”。

[illegible]

1.3.3 命令窗口的显示方式

[illegible]命令和操作设置，希望用户全面了解 MATLAB 的命令系统。

[illegible]对话框，用户可以在其中设置字体格式等，如图 1.6 所示。

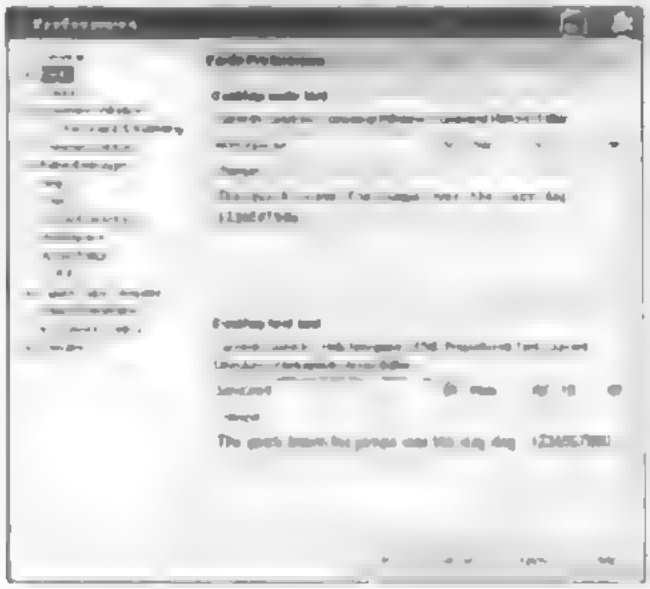


图 1.6 参数设置对话框

[illegible]体属性是 Plain，大小为 10。

[illegible]置对话框中的“OK”按钮，完成属性的设置。

[illegible]

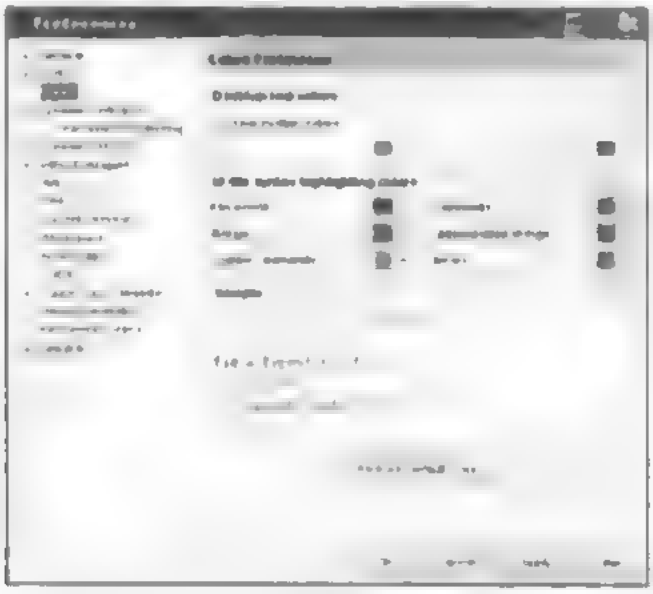


图1.7 设置字体颜色

字体颜色。

必不同类型的字符颜色。

默认的字体颜色。

1.3.4 数值结果的显示方式

算符号自行设置数值结果的显示方式。

表 1.2 数据显示方式的常见命令

命令	说明	举例
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]个小数值，其中位有效数字的科学记数法来显示	[illegible]
format long	15 位数字表示	2.563753324578901
format short e	5 位科学记数表示	2.5638e+00
format long e	15 位科学记数表示	2.563753324578901e+00
format short g	从 format short 和 format short e 中选择最佳的记数方式	2.5638
[illegible]	[illegible] format short 和 format short e 中选择最佳的记数方式	[illegible]
format rat	用近似有理数表示	357911/140
format hex	用十六进制数表示	400482911a609f08
format bank	使用金融数据	2.56

[illegible]，由于[illegible]直接在 MATLAB 命令窗口输入相应命令[illegible]，查看[illegible]a 的不同显示结果，如图 1.8 所示。

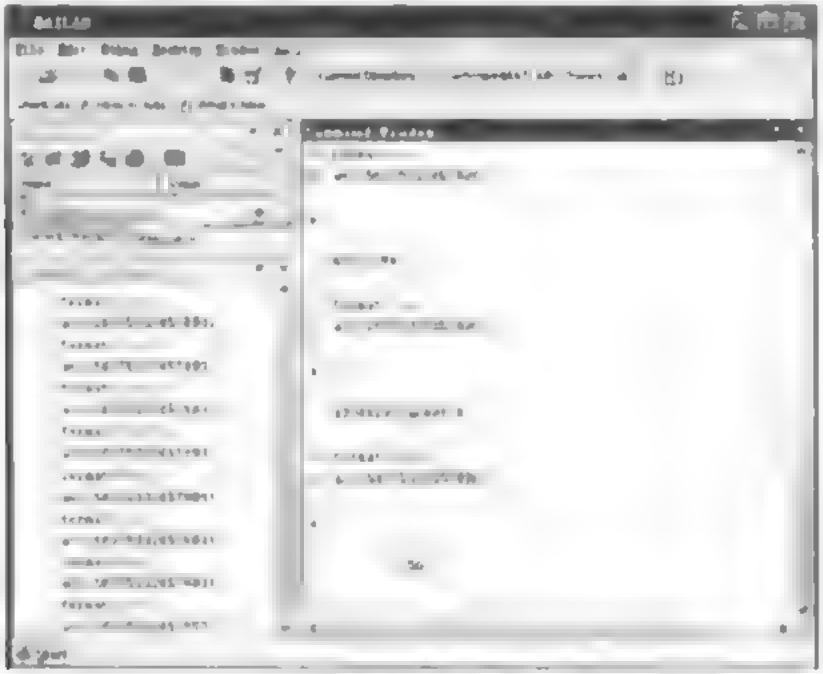


图 1.8 设置数值的显示格式

[illegible]，[illegible]示出不同的结果。

[illegible]

1.3.5 命令窗口的标点符号

[illegible]

表 1.3 MATLAB 常见标点的功能

名称	标点	作用
[illegible]	[illegible]	[illegible]
分号	;	用作命令的结束表示，同时不显示结果；数组元素的行与分隔符
冒号	:	用来生成一维数值数组
逗号	,	输入变量之间的间隔；数组元素的分隔符
小数点	.	数值中的小数点
续行符号	…	用在数据行的开头，表示该数据行是非执行的注释行
方括号	[ ]	输入数组的时候用
[illegible]	[illegible]	[illegible]
		整体

下面列举一些简单的例子，介绍常用标点的功能。

例 1.1 在 MATLAB 中输入数据。

具体的输入步骤如下

Step 1 在 MATLAB 的命令窗口中输入下列内容

[illegible]

Step 2 [illegible] 键，[illegible]

图 1.9 输入数值矩阵

[illegible]

例 1.2 在 MATLAB 命令窗口中输入下面的换行命令

```
B=1 + 25 - 36 +.....
37 - 58 + 77
```

[illegible] 键，[illegible]

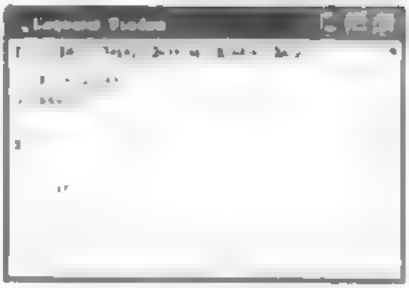


图 1.10　续行输入命令行

[illegible]

例 1.3　在 MATLAB 中进行数组点乘。

具体的输入步骤

Step 1　[illegible]

```
>> %例1.3.3-3演示标点符号的用法
>> [1,2,3].*[4,5,6]
```

Step 2　[illegible]

图 1.11　MATLAB 中的点乘

[illegible]

[illegible]

1.3.6 输入变量

[illegible]

在 MATLAB 中，为变量定义名称需要满足下列规则。

- [illegible]

[illegible] function "EXP"，表明MATLAB无法识别EXP的函数名称，如图1.12所示。

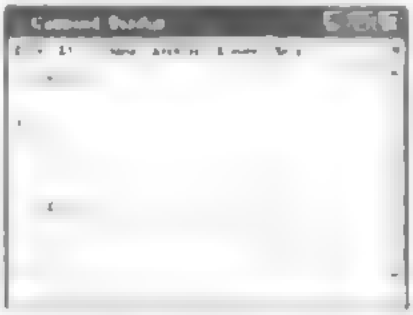


图1.12 函数名称区别大小写

- [illegible] matrix 等都是不合法的变量名。
- [illegible] 是合法的。但是变量名称Num、Var则是不合法的。

[illegible]

[illegible]，表1.4列出了常见的预定义变量名称。

表1.4 MATLAB中的预定义变量

预定义变量	含义
ans	计算结果的默认名称
eps	计算机的零误差
inf（Inf）	无穷大
pi	圆周率
NaN（nan）	表示结果或者变量不是数值

[illegible]

例1.4 [illegible]

上面的详细程序清单如下

```
>> %演示用户重新定义预定义变量
>> pi                          %显示系统的预定义变量pi

ans =

        3.1416
>> R=6;                        %定义半径
>> perimeter=2*pi*R            %计算周长

perimeter =
```

```
        37.699
>> pi=3.50;                    % 重新定义变量 pi
>> perimeter=2*pi*R            % 重新计算周长

perimeter =

    [illegible]
>> clear;                      % 清除用户定义的变量 pi 和 R
>> R=6;                        % 定义半径
>> perimeter=2*pi*R            % 重新计算周长

perimeter =

        37.642
```

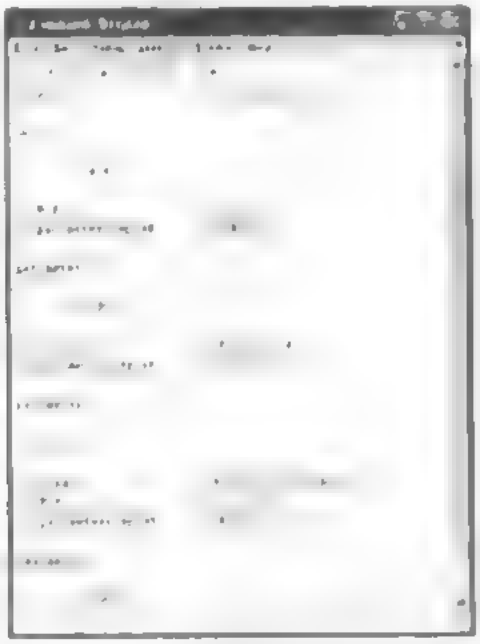


图 1-13　在 MATLAB 中使用预定义变量

[illegible]义变量 pi 的数值。

[illegible]

[illegible]

1.3.7 处理复数

[illegible]

例 1.5 [illegible]

具体的输入步骤如下

step 1 在 MATLAB的命令窗口中输入下列内容

```
>> % 显示如何在MATLAB中输入复数
>> z1=8+10i;                    % 直接使用直角坐标的方式输入
>> z2=12+6*i;                   % 运算符构成的直角坐标的方式输入
>> z3=25*exp(i*pi/3);           % 运算符构成的极坐标的方式输入
>> A=[z1,z2,z3]
```

step 2 按"Enter"键，[illegible]

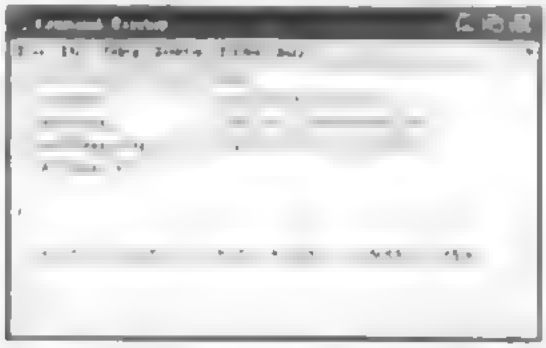


图 1.14 在 MATLAB 中输入复数

[illegible]

例 1.6 在 MATLAB 中输入复数矩阵并进行矩阵运算。

具体的输入步骤如下

step 1 [illegible]

```
>> % 显示如何使用复数矩阵
>> A=[1,3,5;7,9,11]+[2,4,6;8,10,12]*i;          % 使用数据输入复数矩阵
>> B=[1+2*i,3+4*i;5+6*i,7+8*i;9+10*i,11+12*i];   % 使用元素输入复数矩阵
[illegible]
```

Step 2 [illegible]

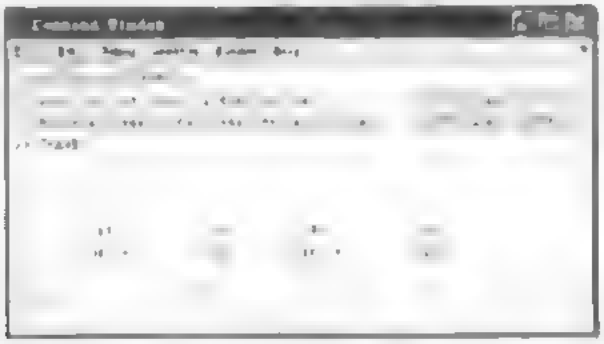


图 1.15　在 MATLAB 中输入复数矩阵

[illegible]

[illegible]

例 1.7　[illegible]

具体的输入步骤

Step 1 [illegible]

```
%处理复数参量（实部、虚部、模和相角）
Real=real(C);      Imag=imag(C);                 %计算复数的实部、虚部
Mag=abs(C);        Phase=angle(C)*180/pi;        %计算复数的模、相角
```

Step 2 [illegible] 1.16 所示。

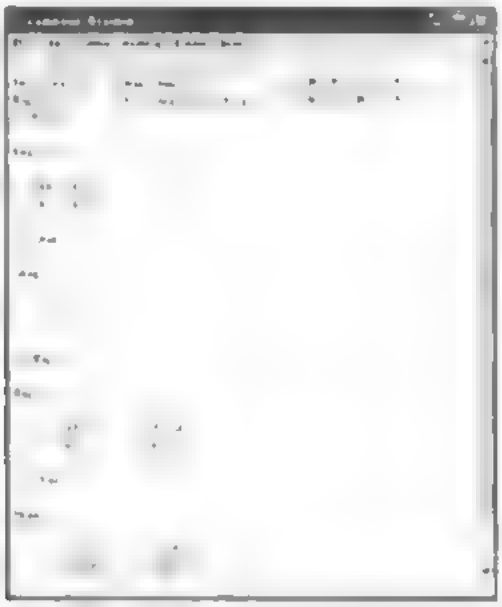


图 1.16　在 MATLAB 中计算复数的参量

[illegible]

1.3.8 命令窗口的控制命令

[illegible]在本小节中归纳了关于MATLAB常见的控制命令和对应功能，如表1.5所示。

表1.5 MATLAB中的常见控制命令

命令	说明
clf	清除图形窗
clc	清除命令窗口中的显示内容
type	显示指定M文件的内容
clear	清除MATLAB工作空间中保存的变量
[illegible]	[illegible]

[illegible]会清除变量。

[illegible]

1.3.9 使用历史窗口

[illegible]个命令。

[illegible]方的按钮，查看输入的历史窗口，如图1.17所示。

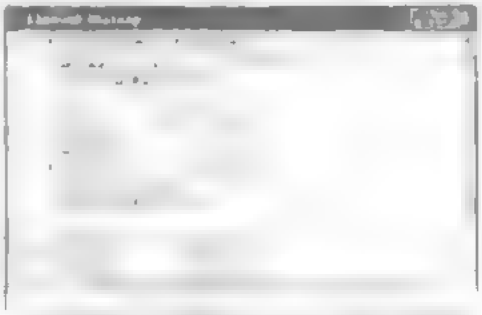


图1.17 历史窗口

历史窗口的菜单栏和命令窗口的菜单栏相同。如果用户希望将历史窗口脱离主MATLAB的操作界面中，可以选择历史窗口中的"Undock"或"Dock Command Window"命令，也可以直接单击菜单栏中的按钮。

在历史窗口中，记录着由在MATLAB命令窗口中输入的所有命令，在非用户主动删除之前，历史窗口中的内容将一直存在。默认情况下，历史窗口中显示的内容是由用户启动MATLAB的时间，包括启动MATLAB的所有命令行。

用户不仅能在历史窗口中查看的命令窗口中曾经执行过的命令，而且还可以根据需要编辑这些命令行。下面列举几个常见的编辑功能。

- **复制命令行：**[illegible]，例如，由于需要输入相似的命令行，有时可以在历史窗口中复制一个命令行，然后单击鼠标右键，在弹出的快捷菜单中选择"Copy"选项，即可[illegible]。

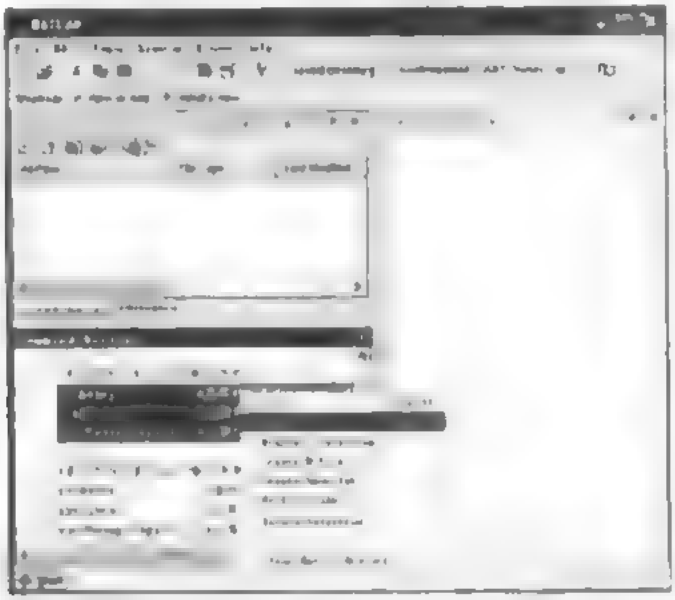


图1.18 复制历史命令行

复制历史命令行后，可以在命令窗口中执行粘贴操作，如图1.19所示。

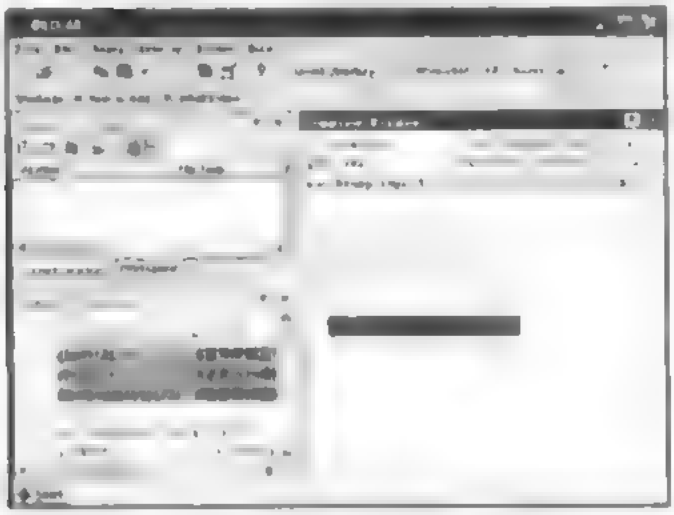


图1.19 粘贴历史命令行

[illegible]

[illegible]

- **运行命令行**：[illegible] Selection" 选项，如图 1.20 所示。

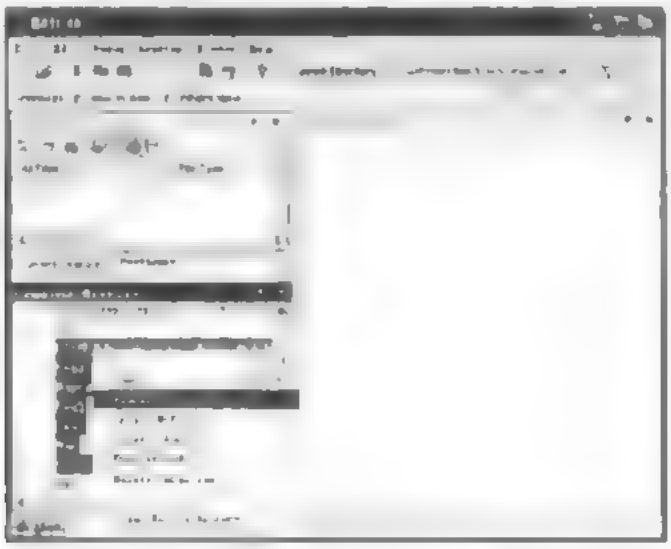


图 1.20 运行历史命令行

[illegible]

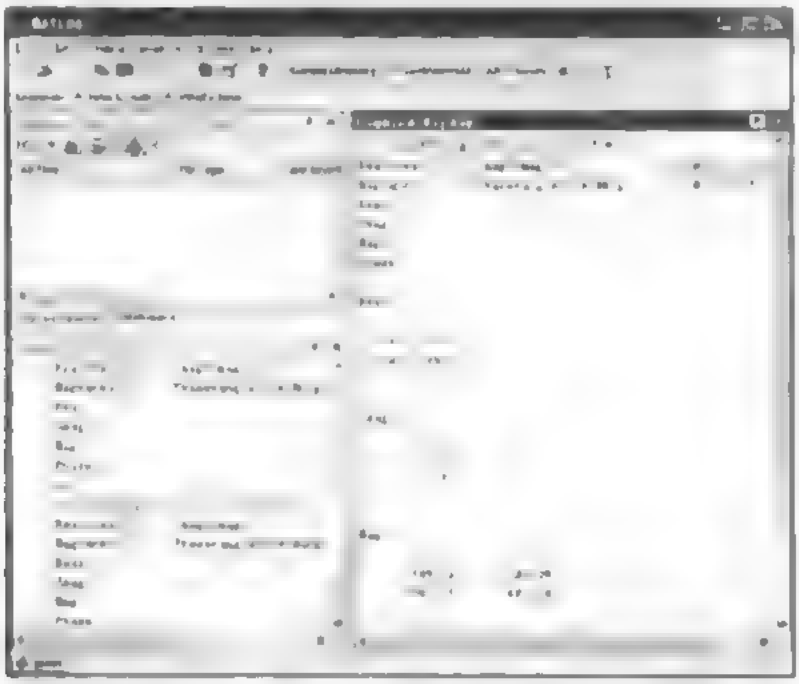


图 1.21 运行的结果

[illegible]

◆ **创建M文件**：[illegible]，现在需要将上述命令编辑成一个M文件。在工具栏上选择需要运行的一个命令，然后单击鼠标右键，在弹出的快捷菜单中选择"[illegible]"选项，如图1.22所示。

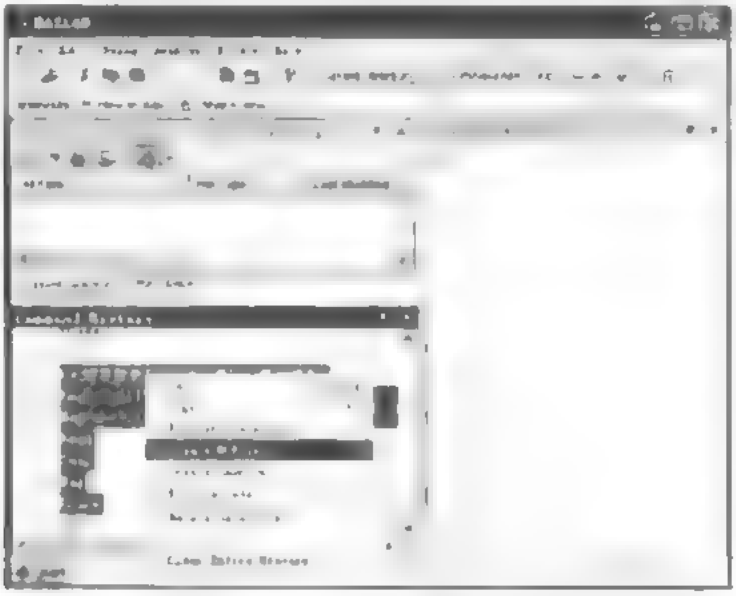
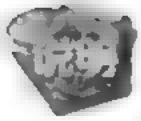


图1.22 创建M文件

[illegible]，MATLAB的文本编辑器，[illegible]，[illegible]在M文件编辑器中，如图1.23所示。

图1.23 创建完成的M文件



[illegible]

1.3.10 使用实录命令

[illegible]用“记事本”或者其他文本软件来阅读日志文件。

[illegible]详细介绍。

例1.8 [illegible]所示。

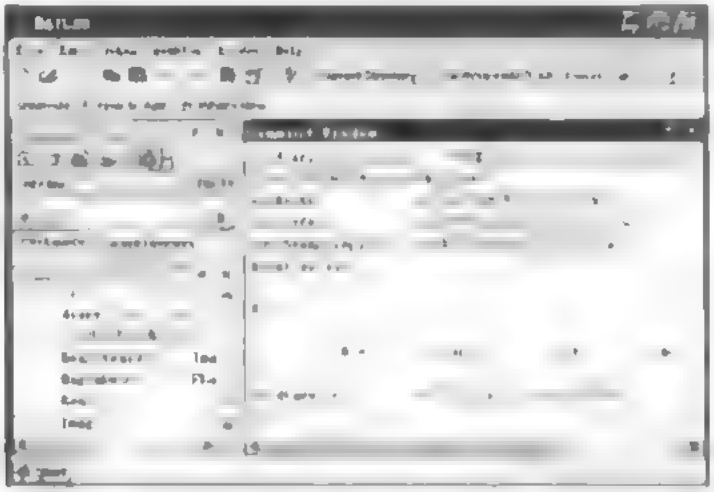


图1.24 创建日志文件

上面的详细程序清单如下

```
>> diary first_diary     %创建名称为first_diary的日志文件
>> %显示如何在MATLAB中输入复数
z1=8+10i;                %直接按照直角坐标的方式输入
z2=12+6*i;               %运算符构成的直角坐标的方式输入
z3=25*exp(i*pi/3);       %运算符构成的极坐标的方式输入
[illegible]

A =

        8 +        10i          12 +          6i        12.5 +      21.6511
>> diary off          %关闭记录命令，完成日志文件
```

[illegible]“记事本”软件打开该日志文件，如图1.25所示。

[illegible]

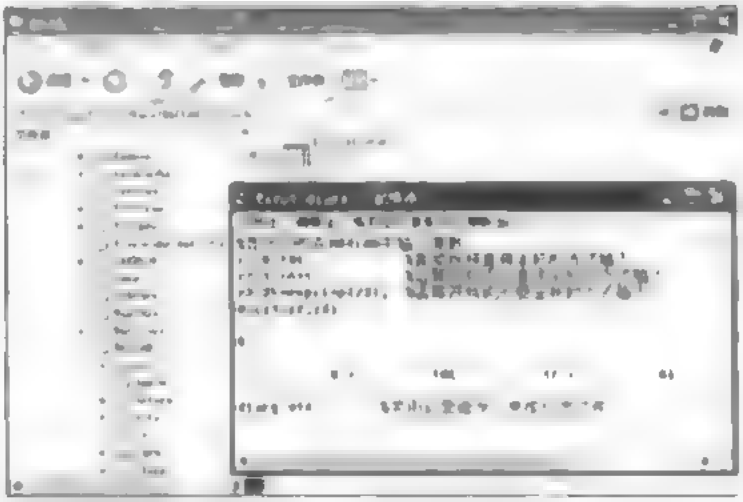


图 1.25　阅读日志文件

1.3.11　当前目录管理器和路径管理

在MATLAB工作空间内的函数和数据库。当用户在命令窗口中输入命令的时候，MATLAB如何搜索到相应的函数和数据，怎样有效地管理这些函数和数据库的路径，提高搜索的效率是本小节的核心问题。

通过，用户在使用MATLAB的时候，中会产生大量的MATLAB文件，如何管理好这些文件也是个重要的问题。如果不能有效管理这些文件，将会直接影响用户在MATLAB的效率。

在MATLAB中，提供当前目录浏览器来管理各种文件。在默认的情况下，当前目录浏览器在MATLAB操作界面的左侧，单击"Current Directory"选项卡，可以使目录浏览器在MATLAB的前台显示。然后单击目录浏览器右上角的▼按钮，可以查看目录管理器的详细外观，如图1.26所示。

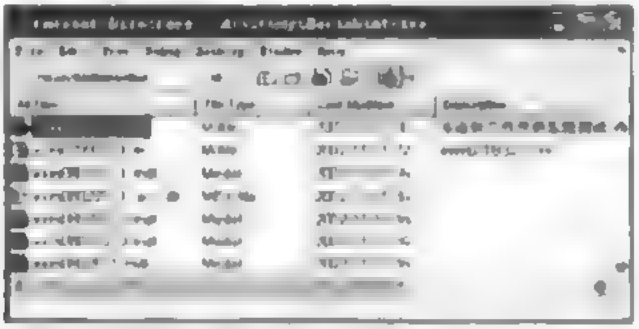


图 1.26　当前目录浏览器

在默认情况下，当前目录浏览器由菜单栏、当前目录设置区、工具栏、单机文件详细列表等。其中，用户需要对菜单栏和文件详细列表区域。在该区域中，用户可以运行或者编辑M文件，装载MAT数据文件等常见操作，如图1.27所示。

上面的操作过程十分简单，用户首先在当前目录浏览器中选择相应的M文件，然，单击鼠标右键，在弹出的快捷菜单中选择对应的选项，例如，用户希望运行该的M文件，可以选择"Run"选项。用户希望编辑该M文件，则选择菜单中的"Open"选项，该M文件就会出现在M文件编辑器中。其他的操作都可以在该菜单中选择对应的选项。

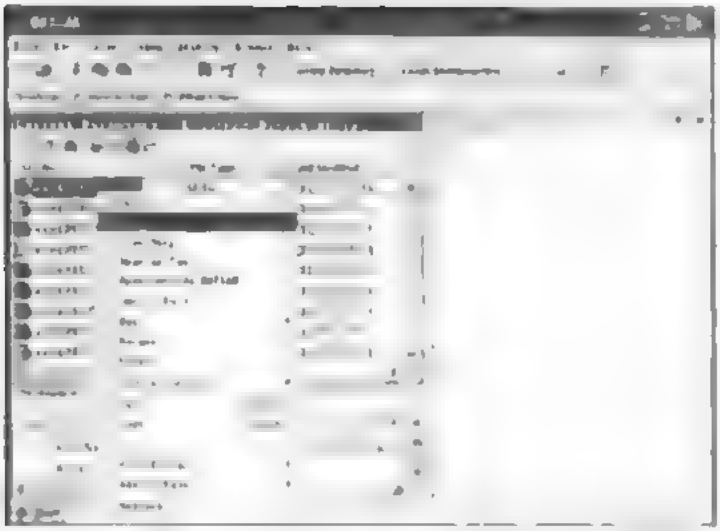


图1.27 操作和编辑M文件

1.3.12 设置当前目录

在默认情况下，启动MATLAB的时候，系统会将当前目录设置为"MATLAB/work"或者"MATLAB/toolwin32"，这取决于用户启动MATLAB的方式，这在前面的内容中已经介绍。

由于"MATLAB/work"这个目录路径，用户在该路径中存放用户的文件是安全的，因此用户可以使用这个默认路径。而对于"MATLAB/toolwin32"这个目录路径，则不建议用户使用，如果用户启动MATLAB系统时候产生"这个路径，建议用户改变这个默认的目录路径。

尽管用户可以采用"MATLAB/work"这个目录路径，但是根据笔者经验，为了方便用户管理各种MATLAB文件，还是建议用户创建自己的工作路径，存放自己创建的应用文件。而将"MATLAB/work"这个目录路径作为临时目录使用。

创建工作目录的方法和在Windows中创建目录的方法完全相同，用户可以参照相应的步骤。建议将用户创建的工作目录设置为当前目录，这是因为在MATLAB环境中，如果不特别指明存放目录，MATLAB都会默认地将文件存放在当前目录。如果用户将自己设置的工作目录设置为当前目录，就可以保证MATLAB运行的可靠和便捷。

用户可以在当前目录浏览器中的"目录设置栏"中输入新的工作目录，或者单击该界面中的"目录浏览键"，选择新的工作目录，如图1.28所示。

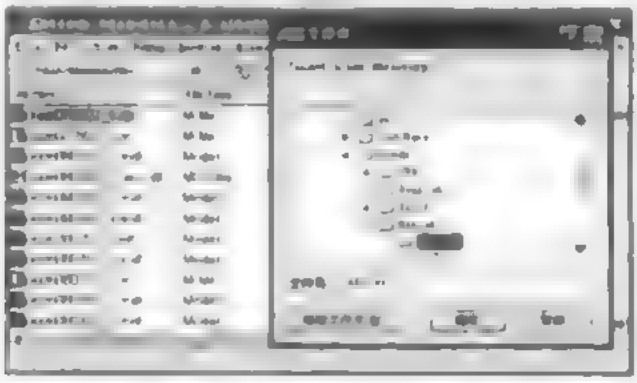


图1.28 设置当前目录

新工作目录的方法，可以在"命令行窗口"中输入相应的命令语句来改变当前目录，这

[illegible] mfiles。对应的控制命令为 cd D:\Study\Matlab\mfiles。

[illegible]

1.3.13 MATLAB 的搜索路径

[illegible] 进行搜索，查找相应的文件、函数或者具体的数据。

[illegible] mat lab [illegible] mat lab 的 M 文件[illegible] mat lab 的 M 文件。

[illegible] 管理，指定搜索次序和 MATLAB 的运行效率。

[illegible] MATLAB 的搜索路径中。

[illegible] 命令，[illegible] 命令，[illegible]

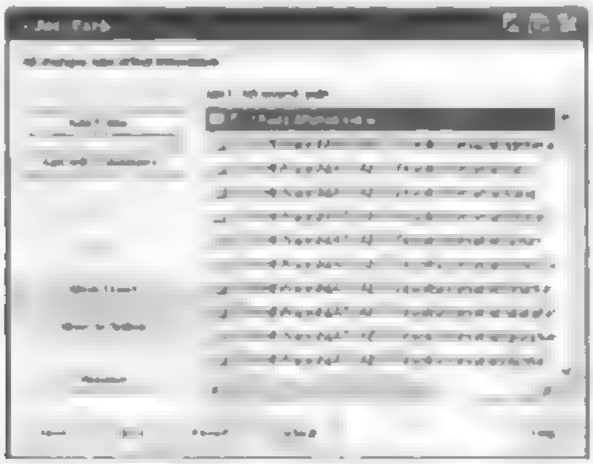


图 1.29 路径设置对话框

[illegible] 按钮。

[illegible]

[illegible]。例如，可以使用path命令查看MATLAB的路径信息，如图1.30所示。

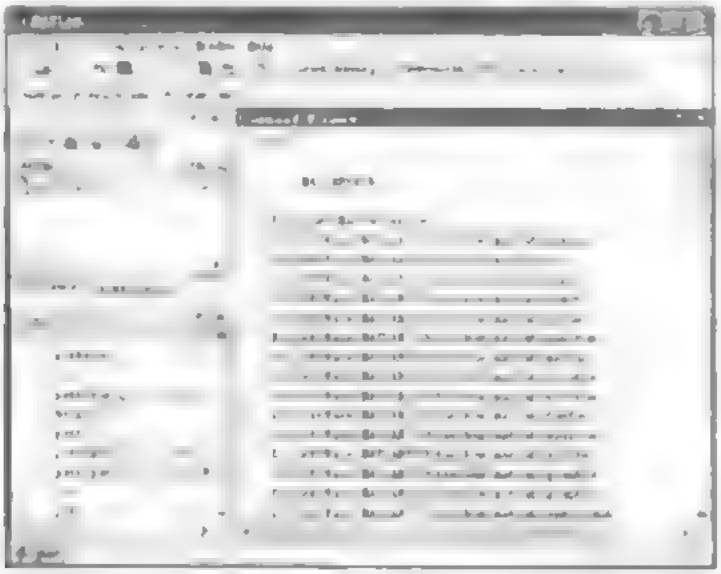


图1.30 查看MATLAB的路径信息

[illegible]

1.3.14 工作空间浏览器和数组编辑器

[illegible]工作空间浏览器的详细外观，如图1.31所示。

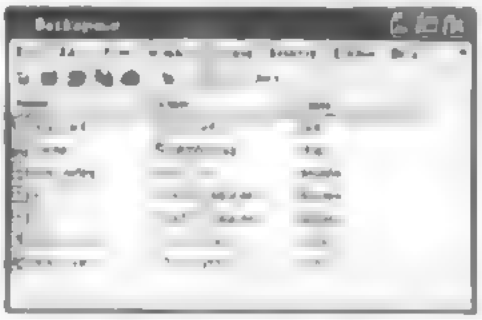


图1.31 工作空间浏览器

和其他MATLAB软件的界面设计，[illegible]

1.32所示。

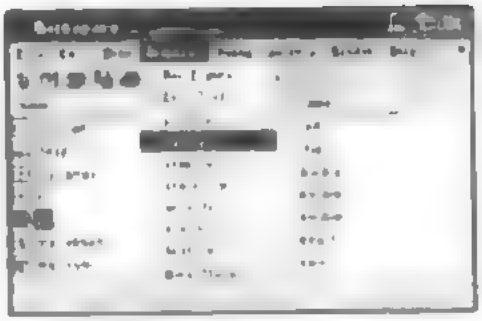


图1.32 图形选项菜单

在[illegible]面对[illegible]表类型，用户可以选择菜单中的"More Plots"选项，打开"Plot Catalog"对话框，选择所需的图表类型，如图1.33所示。

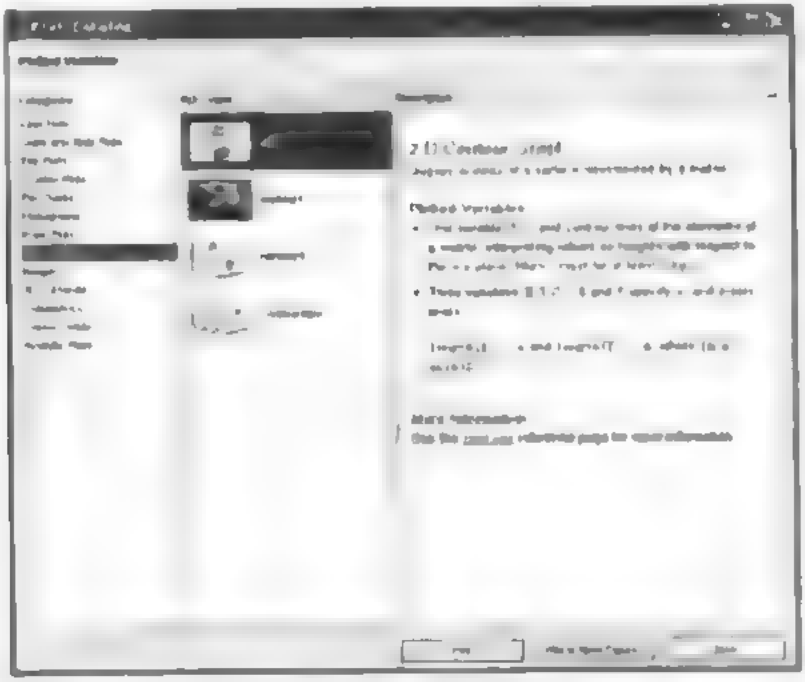


图1.33 图形分类对话框

上面的"Plot Catalog"对话框中的左侧是图形的分类，中间表示的是图表类型，右侧则是用户选择的图表类型的描述。

[illegible]

除了非常明显的图形绘制功能之外，工作空间浏览器还有其他多项实用功能，例如对变量的查看、保存和编辑等，所有这些操作工作都在菜单，只需要在工作空间浏览器中选择相应的变量，然后右击鼠标，在弹出的快捷菜单中选择相应的菜单选项，如图1.34所示。

对变量的操作工作可以从菜单选项中的名称看出，例如，"Rename"菜单选项表示对应的操作是重命名该变量，"Copy"菜单选项表示对应的操作是复制该变量等。

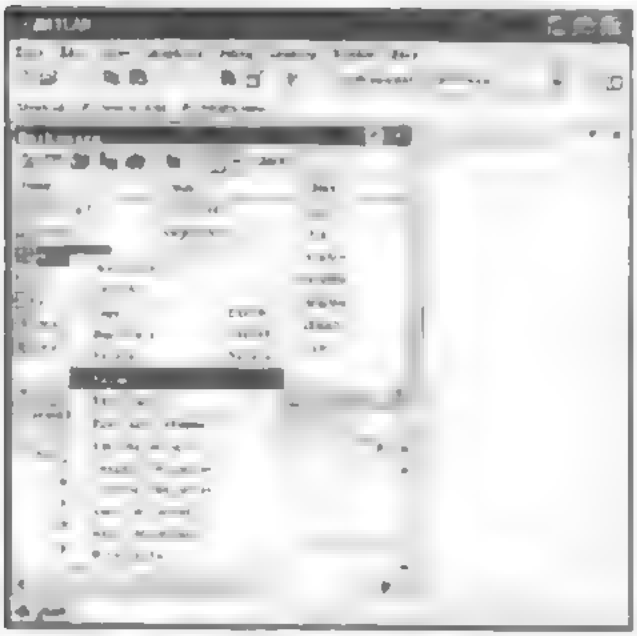


图 1.34 编辑内存变量

1.3.15 变量的编辑命令

在 MATLAB 中，[illegible]外，[illegible]在 MATLAB 的命令窗口中输入生成[illegible]，[illegible]。[illegible]对变量进行操作。

例 1.9 在 MATLAB 命令窗口中查阅内存变量。

具体步骤如下：

在命令窗口中输入 who 或 whos 命令，查看内存变量的信息，[illegible]。

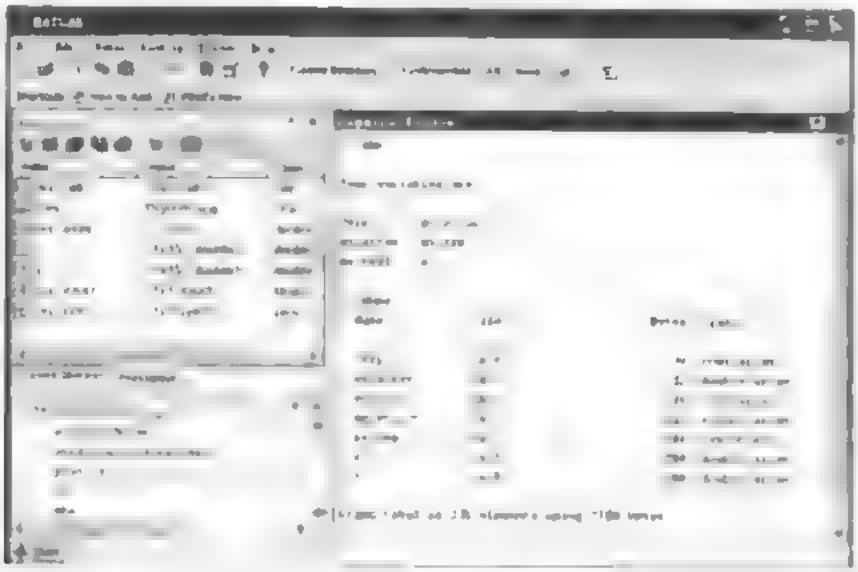


图 1.35 查阅内存变量的信息

[illegible]

例 1.10　[illegible]，在 MATLAB 命令窗口中删除内存变量 Desp。

在命令窗口中输入下面的命令行

```
>> clear Desp;
>> who
```

[illegible] 1.36 所示。

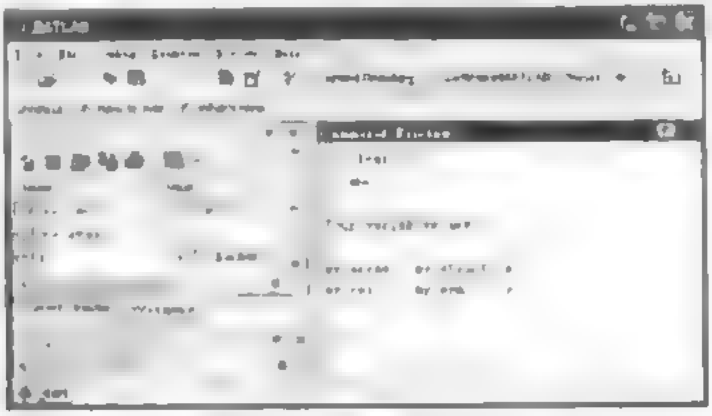


图 1.36　删除内存变量

[illegible]，用户可以看出，当[illegible] clear 命令后，将 Desp 变量从工作空间删除，而且在工作空间浏览器中也将该变量删除。

如果[illegible]在命令窗口中直接输入 clear 命令，就可以删除工作空间中的所有变量。如果用户需要删除某个内存变量，可以在 clear 命令后面跟上要删除的变量名称。

1.3.16 数组编辑器

在 MATLAB 中，数组和矩阵都是十分重要的基础变量，因此 MATLAB 专门提供数组编辑器这个工具来编辑数组。选择工作空间浏览器中任意一个数组（就是 class 类别为 double 的内存变量），然后单击菜单栏中的"Open Selection"选项，或者直接双击该变量，就可以打开该变量的数组编辑器，如图 1.37 所示。

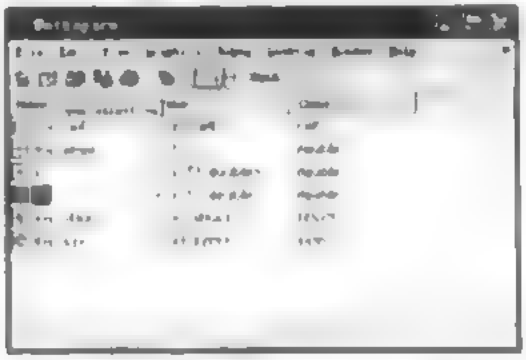
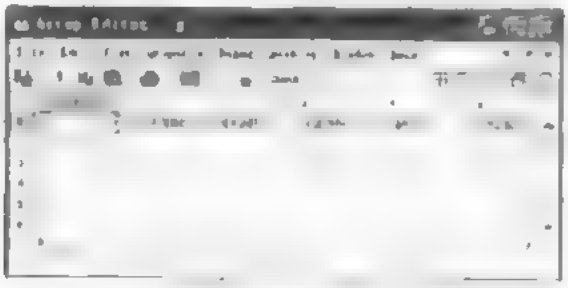


图 1.37　打开数组编辑器



说明

在MATLAB中，数组编辑器[illegible]，[illegible]数值数组打开相应的数组编辑器。

在[illegible]，[illegible]变量的数组编辑器，如图1.38所示。

图1.38 变量的数组编辑器

[illegible]在数组编辑器中[illegible]编辑该变量，[illegible]的便利。

1.3.17 存取数据文件

在MATLAB中，提供了save和load命令来实现数据文件的存取，表1.6[illegible]。

表1.6 MATLAB文件存取的命令

命令	功能
save filename	[illegible]
save filename x y z	[illegible]
save filename -regexp pat1 pat2	[illegible]
save filename x y z -ASCII	[illegible]文件中
load filename	[illegible]
load filename x y z	[illegible]
load filename -regexp pat1 pat2	[illegible]
load filename x y z -ASCII	[illegible]

[illegible]，用户可以根据需要选择[illegible]。[illegible]较少见的存取命令，用户可以查阅MATLAB的相关帮助。

说明

[illegible]式来处理，生成以MAT扩展名的文件。

在MATLAB中，除了可以在命令窗口中输入相应的命令之外，也可以在[illegible]，实现数据文件的存取工作。例如，用户可以选择[illegible] Save Workspace

save 命令，将所有变量保存到 mat 文件中，如图 1.39 所示。

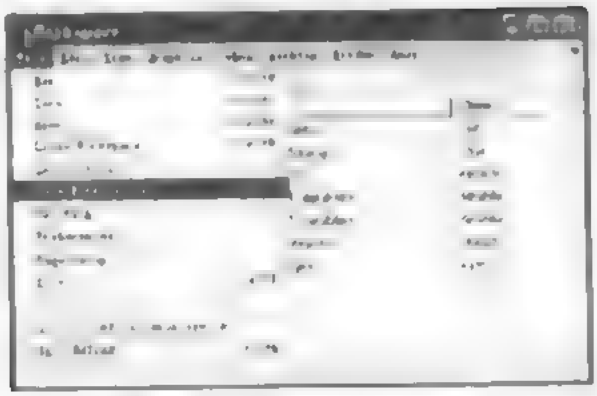


图 1.39 保存所有的变量

如果选择"save Workspace As"命令，则会弹出"Save to MAT-File"对话框，用户可以在文件名文本框中输入新的文件名称，并选择保存路径，将工作空间中的变量保存。

如果用户需要保存部分变量，可以在工作空间窗口中选择需要保存的变量，然后单击鼠标右键，在弹出的快捷菜单中选择"save As"命令，将所选择的变量保存到 mat 文件中，如图 1.40 所示。

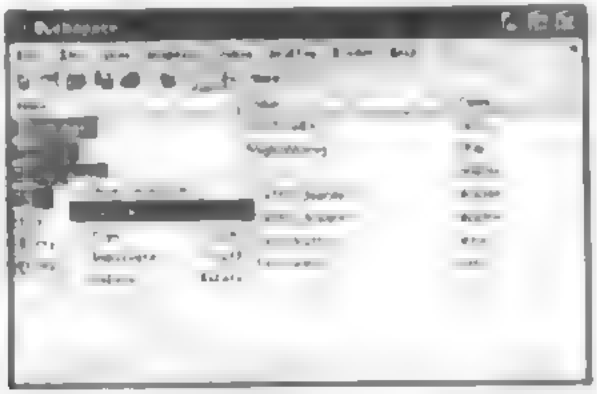


图 1.40 保存部分变量

选择"save As"命令，弹出"save MAT-File"对话框，用户可以在文件名文本框中输入新的文件名称，并选择保存路径，将工作空间中选择的变量保存。

在 MATLAB 的工作空间窗口中，用户可以导入数据，选择"File"菜单中的"Import Data"命令，弹出"Open"对话框，选择要导入的数据文件，单击"打开"按钮，打开"Import Wizard"对话框，用户可以在打开的"Import Wizard"对话框中选择希望导入的变量，如图 1.41 所示。

在"Import Wizard"对话框中，左侧表明了数据文件中的变量，用户可以在此变量中选择需要导入的变量，对话框右侧显示选定变量的预览效果。在图中，用户选择文件变量"my_array"，在右侧显示该变量的预览效果。

如果某个数据文件中的变量名称与工作空间中的变量名称相同，则会覆盖工作空间中的变量，建议使用 who 命令查看工作空间中的变量名称，以免发生冲突。用户也可以使用 clear 命令清除工作空间中的变量。

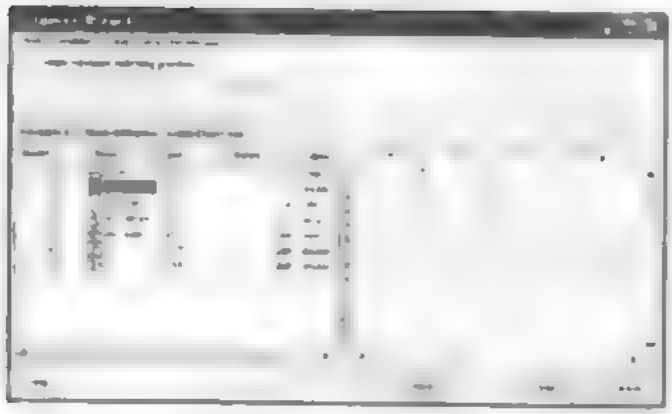


图 1.41 加载变量的对话框

1.4 MATLAB7.0的帮助系统

[illegible]

1.4.1 纯文本帮助

[illegible]

例 1.11 如何在MATLAB中查阅帮助信息。

[illegible]

step 1 [illegible] 使用help命令，如图 1.42所示。

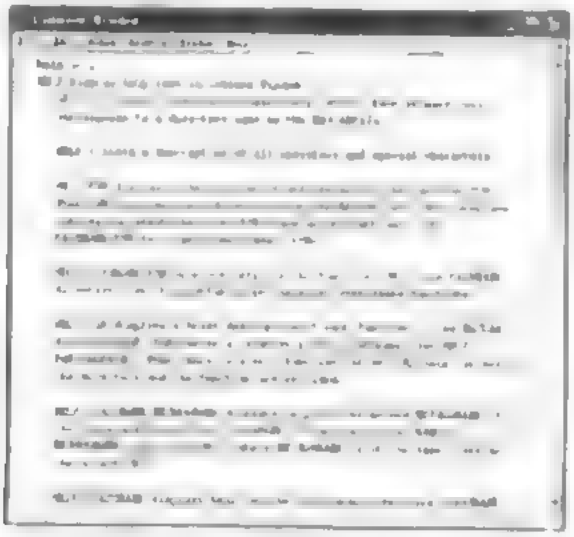


图 1.42 使用help命令的帮助信息

[illegible] 的信息来帮助回使用help命令。

Step 2 [illegible] 有主题的帮助信息，如图 1.43 所示。

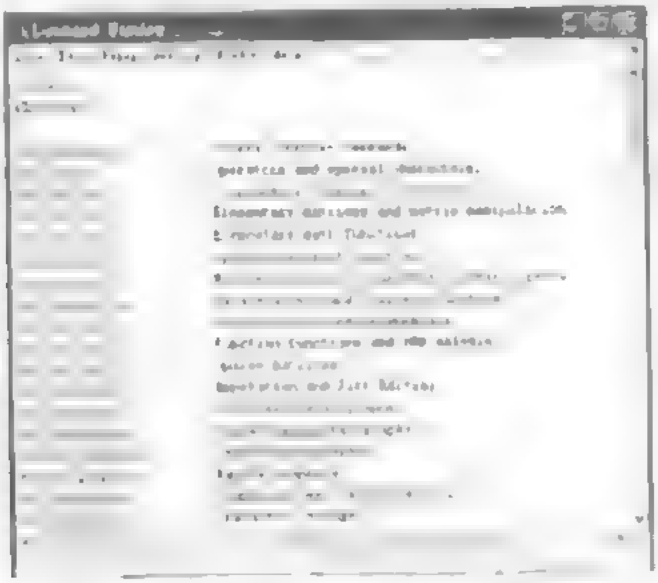


图 1.43 查询关于主题的帮助信息

Step 3 [illegible] 有帮助信息，如图 1.44 所示。

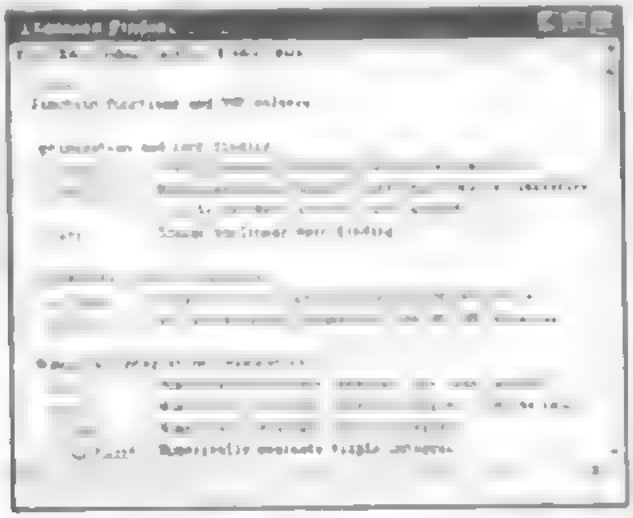


图 1.44 查看主题下的函数帮助信息

[illegible]

例 1.12 [illegible] 的所有M函数文件名，如图 1.45 所示。

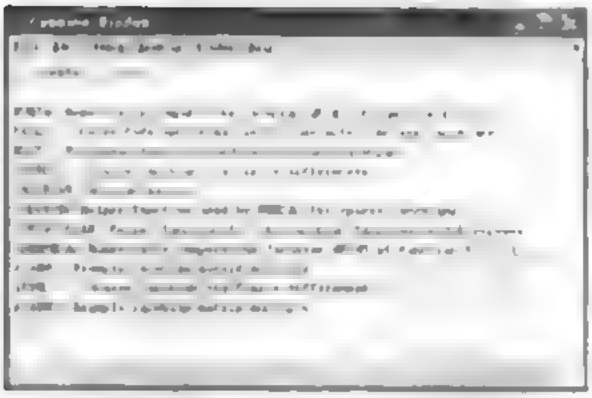


图 1.45 搜索M文件的帮助文件

1.4.2 演示（demo）帮助

在MATLAB中，各个工具箱都有自己的演示程序，[illegible]，操作[illegible]简单、直观，[illegible]MATLAB[illegible]对MATLAB[illegible]要的文件[illegible]演示，不需要[illegible]，[illegible]MATLAB[illegible]掌握的。

在MATLAB的命令窗口中输入“demo”命令，[illegible]，[illegible]1.46所示。

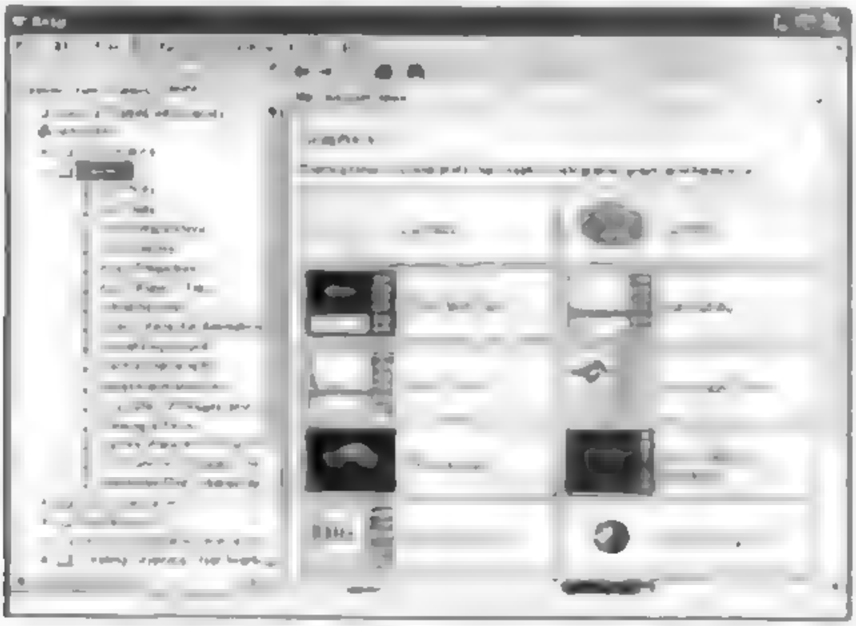


图 1.46 MATLAB中的demo帮助

[illegible]，[illegible]Surface Plots”选项。MATLAB[illegible]“3-D Surface Plots”[illegible]“Run this demo”[illegible]“3-D Plots[illegible]”[illegible]，[illegible]就是演示demo的交互界面，如图1.47所示。

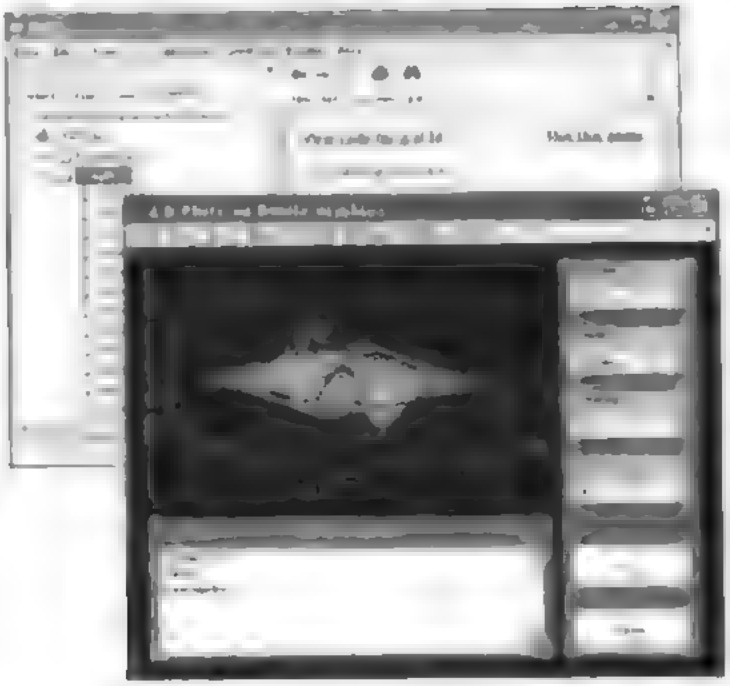


图 1.47　显示 demo 的交互界面

[illegible]

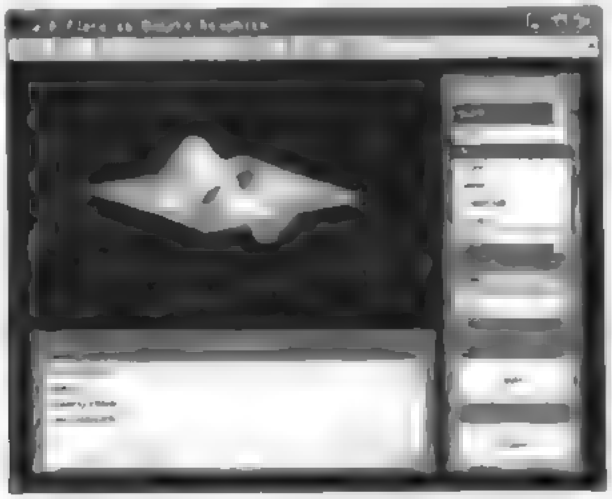


图 1.48　动态演示 (demo)

[illegible]

[illegible]

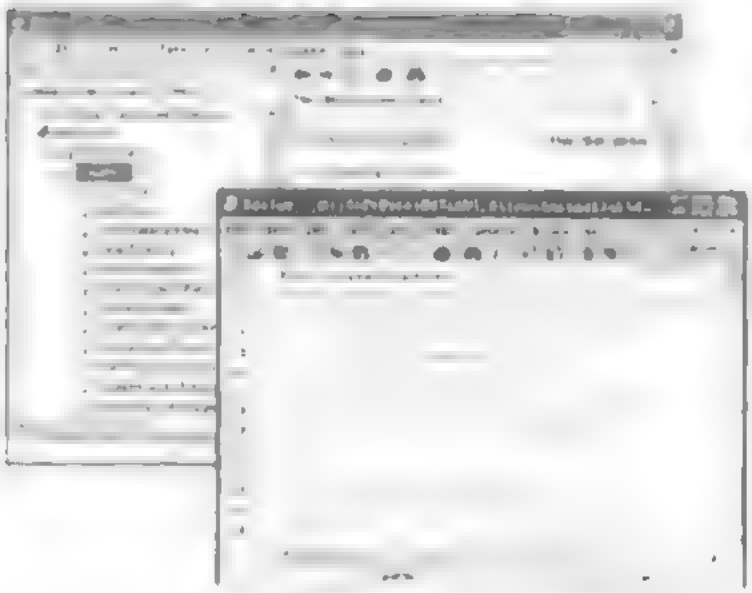


图 1.49 查看 demo 的源序代码

1.4.3 帮助导航/浏览器

[illegible]友好。使用方便，是用户查找帮助的重要途径。

[illegible]

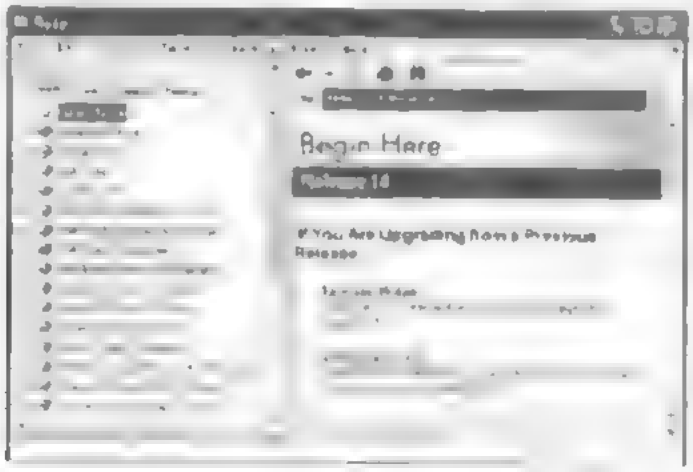


图 1.50 帮助导航/浏览器界面

1.4.4 Contents 帮助文件目录窗

[illegible]条，就可以在浏览器中显示相应标题的 HTML 帮助文件。

[illegible]

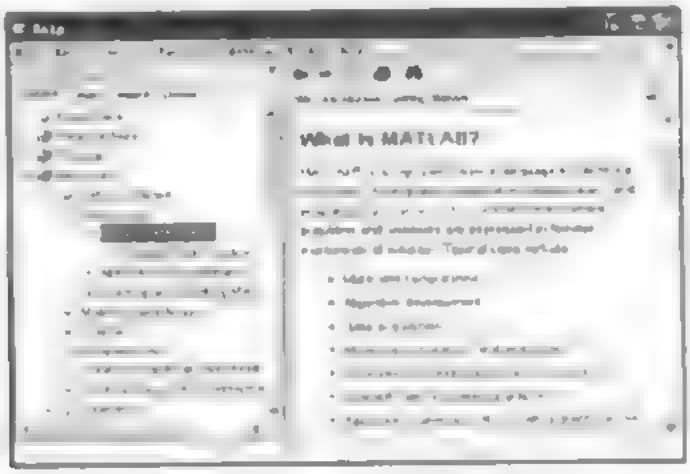


图 1.51　查看帮助文件的目录

[illegible]

1.4.5 Index 帮助文件索引窗

[illegible]

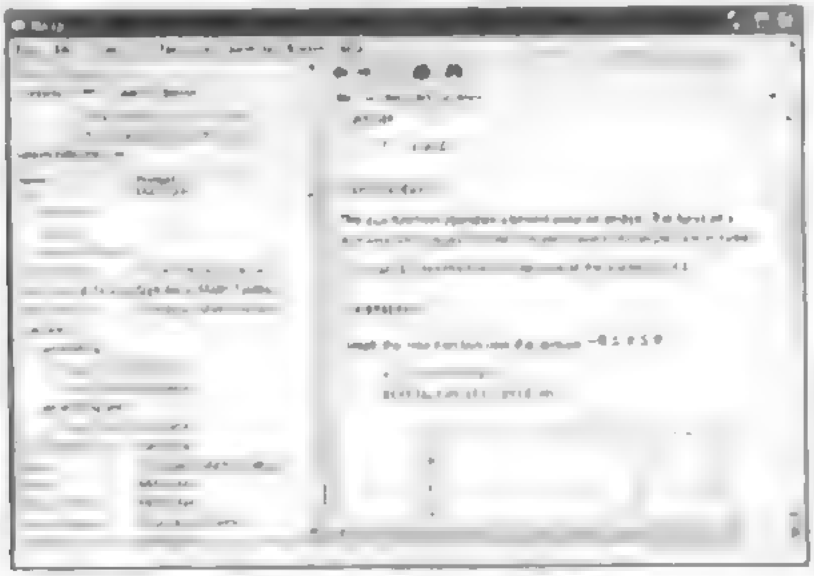


图 1.52　查看"sin"的信息

[illegible]

1.4.6 Search 帮助文件搜索窗

[illegible]

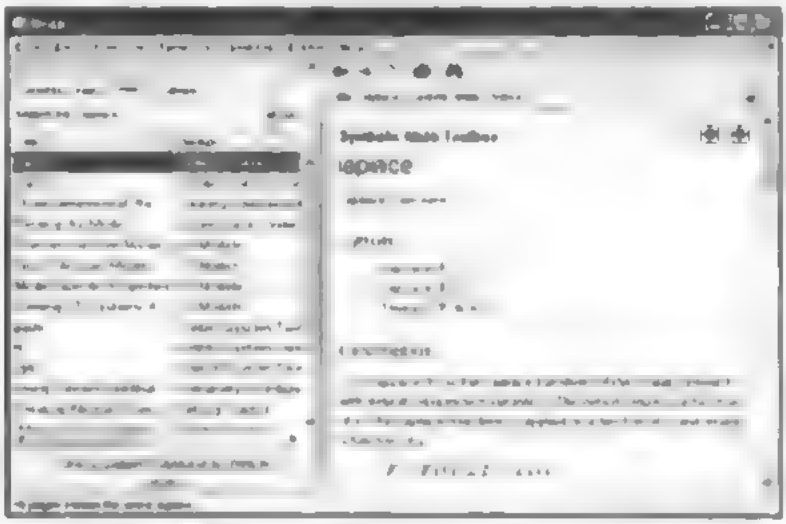


图 1.53 在“Search”窗口中查看“laplace”的信息

在搜索结果窗口底部，有“Search support database on Web for laplace”的超链接文字，单击该文字，MATLAB就会启动互联网浏览器，搜索关于laplace的更多信息。[illegible]，在搜索窗口的最底部，显示“18 pages contain the word laplace”的字样，表示搜索的结果个数。

在帮助浏览器中会显示每个搜索结果的HTML文件，Search关键字“laplace”也会显示，方便用户查阅相应的信息。

[illegible]

1.5 小结

[illegible]

◆ ◆ ◆

第2章 MATLAB基础知识

本章包括

- 创建数值数组
- 稀疏矩阵
- 字符串数组
- 操作数值数组
- 构架数组

数组或者矩阵都是MATLAB的基础内容，几乎所有的数据都是用数组的形式进行储存的，因此，MATLAB又被称为矩阵实验室。从MATLAB5.x版本开始，基于面向对象的考虑，这些数组就成为MATLAB中的内建数据类型(Built-in Data Type)，而数组运算就是定义这些数据结构上的方法。在本章中，将介绍关于数组类型和数组运算的内容。

创建数值数组

创建数组是所有MATLAB运算和操作的基础。针对不同维度的数组，MATLAB提供多种创建方法，可以分别创建不同要求的数组类型。在本节中，将分别根据数组维度以及方法的不同，介绍如何创建数组。

一维数组的创建方法

在MATLAB中，一维和二维数组都被认为是比较低维的数组。它们的创建方法比较简单，同时，也是创建高维数组的基础条件，在本小节中，将以简单的例子来说明如何在MATLAB中创建各种不同的数组类型。

例2.1 在MATLAB中，使用不同的方法来创建一维数组。

step 1 在MATLAB的命令窗口中输入下面的程序代码：

```
>> data1=[pi;log(5);7+6;2^3];
>> data2=[pi log(5) 7+6 2^3];
>> data3=2:2:10;
>> data4=2:10;
>> data5=linspace(2,10,5);
>> data6=logspace(1,5,10);
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果：

```
data1 =
    3.1416
    1.6094
   13.0000
    8.0000
data2 =
    3.1416    1.6094   13.0000    8.0000
```

```
data3 =
     2     4     6     8    10
data4 =
     2     3     4     5     6     7     8     9    10
data5 =
     2     4     6  [illegible]
data6 =
  1.0e+005 *
    0.0001    0.0003    0.0008    0.0022    0.0060    0.0167    0.0464
0.1292    0.3594    1.0000
```

上面的结果基本演示了在 MATLAB 中创建一维数组的方法。

- ◆ **直接输入法：** [illegible]了一维行数组。
- ◆ **步长生成法：** [illegible]
- ◆ **定数线性采样法：** [illegible]
- ◆ **定数对数采样法：** [illegible]式为 x=logspace(a,b,n)。

说明 [illegible]

2.1.2 二维数组的创建方法

在本小节中，将介绍如何在 MATLAB 中创建二维数组。

例 2.2 在 MATLAB 中创建二维数组。

step 1 在 MATLAB [illegible]

```
>> Data1= [1       2       3
           4       5       6
           7       8       9];
>> Data2=[1,2,3;4,5,6;7,8,9];
```

step 2 [illegible]

```
Data1 =
     1     2     3
     4     5     6
     7     8     9
Data2 =
     1
```

```
     4     5     6
     7     8     9
```

[illegible]

- 整个输入数组必须以方括号“[]”作为创建的首尾
- 数组的行和行之间必须用分号“;”来间隔
- 数组的列和列之间必须用逗号“,”来间隔。

[illegible]

2.1.3 使用下标创建三维数组

在MATLAB中，[illegible]维数组的第一维称为“行”，第二维称为“列”，而对于三维数组，其第三维[illegible]称为“页”。在MATLAB中，将三维或者三维以上的数组统称为高维数组。由于高维数组可以看作是平面数组的堆叠，在本小节中将主要以三维数组为例来介绍如何创建高维数组。

例2.3 使用下标引用的方法创建三维数组。

step 1 在MATLAB的命令窗口中输入一段的程序代码

```
A(2,2,2)=1;
>> for i=1:2
for j=1:2
for k=1:2
A(i,j,k)=i+j+k;
end
end
end
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果

```
>> A(:,:,1)
     3     4
     4     5
>> A(:,:,2)
     4     5
     5     6
```

step 3 创建新的多维数组，在MATLAB的命令窗口中输入下面的程序代码

```
>> B(3,4,:)=2:5;
```

step 4 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果

```
>> B(:,:,1)
     0     0     0     0
     0     0     0     0
     0     0     0     2
>> B(:,:,2)
```

```
     0     0     0     0
           [illegible]     0     0
           [illegible]     0     3
>> [illegible](:,:,3)
           [illegible]     0     0
           [illegible]     0     0
           [illegible]     0     4
>> [illegible](:,:,4)
           [illegible]     0     0
           [illegible]     0     0
```

[illegible]

2.1.4 使用低维数组创建三维数组

在本小节中，将介绍如何在 MATLAB 中使用低维数组创建三维数组。

例 2.4 使用低维数组来创建高维数组。

step 1 在 MATLAB 的命令窗口中输入下面的程序代码

```
>> D2=[1,2,3;4,5,6;7,8,9];
>> D3(:,:,1)=D2;
>> D3(:,:,2)=2*D2;
>> D3(:,:,3)=3*D2;d
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果

```
>>D2
D2 =
     1     2     3
     4     5     6
     7     8     9
>> D3
D3(:,:,1) =
     1     2     3
     4     5     6
     7     8     9
D3(:,:,2) =
     2     4     6
     8    10    12
    14    16    18
D3(:,:,3) =
     3     6     9
    12    15    18
    21    24    27
```

[illegible]

2.1.5 使用创建函数创建三维数组

在本节中，将介绍如何使用MATLAB的创建函数来创建三维数组。

例2.5 使用函数命令来创建三维数组。

Step 1 使用cat命令来创建三维数组。在MATLAB的命令窗口中输入下面的程序代码

```
>> D2=[ 1,2,3;4,5,6;7,8,9] ;
>> C=cat(3,D2,2*D2,3*D2);
```

Step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果

```
C(:,:,1) =
     1     2     3
     4     5     6
     7     8     9
C(:,:,2) =
     2     4     6
     8    10    12
    14    16    18
C(:,:,3) =
     3     6     9
    12    15    18
    21    24    27
```

cat命令的功能是连接数组，其语法格式为C = cat(dim,A1,A2,A3,A4...)。其中，dim表示创建新维度的维度，A1,A2,A3,A4表示新的多维度上的数组。

Step 3 使用repmat命令来创建数组。在MATLAB的命令窗口中输入下面的程序代码

```
>> D2=[ 1,2,3;4,5,6;7,8,9] ;
>> D3 = repmat(D2,2,3);
>> D4=repmat(D2,[ 1 2 3] );
```

Step 4 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果

```
D2 =
     1     2     3
     4     5     6
     7     8     9
D3 =
     1     2     3     1     2     3     1     2     3
     4     5     6     4     5     6     4     5     6
     7     8     9     7     8     9     7     8     9
     1     2     3     1     2     3     1     2     3
     4     5     6     4     5     6     4     5     6
     7     8     9     7     8     9     7     8     9
D4(:,:,1) =
     1     2     3     1     2     3
     4     5     6     4     5     6
     7     8     9     7     8     9
D4(:,:,2) =
     1     2     3     1     2     3
     4     5     6     4     5     6
     7     8     9     7     8     9
```

```
D4(:,:,3) =
     1     2     3     1     2     3
     4     5     6     4     5     6
     7     8     9     7     8     9
```

reshape命令[illegible]

step 5 [illegible]

```
[illegible]
>> D3=reshape(D2,2,2,3);
>> D4=reshape(D2,2,3,2);
>> D5=reshape(D2,3,2,2)
```

step 6 [illegible]

```
>> D2
D2 =
     1     2     3     4
     5     6     7     8
     9    10    11    12
>> D3
D3(:,:,1) =
     1     9
     5     2
D3(:,:,2) =
     6     3
    10     7
D3(:,:,3) =
    11     8
     4     [illegible]
>> D4
D4(:,:,1) =
     1     9     6
     5     7    10
D4(:,:,2) =
     [illegible]    11     8
             4    12
>> D5
D5(:,:,1) =
     [illegible]
     5     6
     9    10
D5(:,:,2) =
     3     4
     7     8
    11    12
```

reshape命令[illegible]

2.1.6 创建低维标准数组

[illegible]

例 2.6 使用标准数组命令创建低维数组。

Step 1 在 MATLAB 的命令窗口中输入下面的程序代码

```
>> A=zeros(3,2);
>> B=ones(2,4);
>> C=eye(4);
>> D=magic(5);
>> randn('state',0);
>> E=randn(1,2);
>> F=gallery(5);
```

Step 2 查看标准结果。在命令窗口输入变量名称，可以得到一个新的标准结果

```
A =
     0     0
     0     0
     0     0
B =
     1     1     1     1
     1     1     1     1
C =
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
D =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
E =
   -0.4326   -1.6656
F =
          -9          11         -21          63        -252
          70         -69         141        -421        1684
        -575         575       -1149        3451      -13801
        3891       -3891        7782      -23345       93365
        1024       -1024        2048       -6144       24572
```

[illegible]

2.1.7 创建高维标准数组

在本小节中，将介绍如何使用标准数组函数来创建高维标准数组。

例2.7 使用标准数组命令创建高维数组。

step 1 在MATLAB的命令窗口输入下面的命令：

```
%设置随机数据源的初始条件
>> rand('state',1111);
>>D1=randn(2,3,5);
>> D2=ones(2,3,4);
```

step 2 查看标准数组，在命令窗口输入变量名称，可以查看生成的结果：

```
>> D1
D1(:,:,1) =
    0.8156    1.2302    1.1906
    0.7119    0.6666    [illegible]
D1(:,:,2) =
   -0.0198   -1.6041   -1.0565
   -0.1567    0.2573    1.4151
D1(:,:,3) =
   -0.8051    0.2193   -2.1707
    0.5287   -0.9219   -0.0592
D1(:,:,4) =
   -1.0106    0.5077    0.5913
    0.6145    1.6924   -0.6436
D1(:,:,5) =
    0.3803   -0.0195    0.0000
   -1.0091   -0.0482   -0.3139
>> D2
D2(:,:,1) =
     1     1     1
     [illegible]
D2(:,:,2) =
     1     1     1
     [illegible]  1     1
D2(:,:,3) =
     1     1     1
     [illegible]  1     1
D2(:,:,4) =
     1     1     1
     [illegible]  1     1
```

[illegible]

2.2.5 操作数值数组

在MATLAB中，除了常用的数组生成、[illegible]，还需要对数组进行各种操作，[illegible]，常用的数组转换操作，MATLAB都提供了大量的函数命令，在本节中将[illegible]。由于数组维度的不同，在MATLAB中的操作方法，[illegible]分别对数组的操作。

2.2.1 选取低维数组的对角元素

[illegible]，使用数组的索引，也可以选取数组的对角元素或生成一个新矩阵。本小节中将首先使用实例来介绍如何操作低维数组。

例2.8 使用diag命令来选取对角元素或者创建矩阵。

step 1 [illegible]

```
[illegible]
[illegible]
[illegible]
[illegible]
B2=diag(A3,1);
```

step 2 [illegible]

```
Data =
     1     2     3     4
     5     6     7     8
     9    10    11    12
A1 =
     [illegible]
    12
A2 =
     [illegible]
    11
A3 =
     [illegible]
B1 =
     0     1     0     0
     0     0     6     0
     0     0     0    11
     0     0     0     [illegible]
B2 =
     0     5     0
     0     0    10
     0     0     0
```

[illegible]

[illegible]

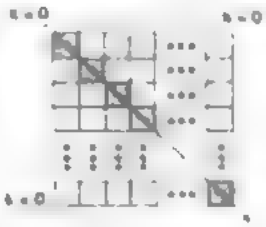


图2.1 diag命令中参数k的含义

低维数组的形式转换

例 2.9 对数组或者矩阵来进行形式转换：对称变换和旋转。

step 1 在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> Data=[1,2,3,4;5,6,7,8;9,10,11,12];
%矩阵的转置
>> B=Data';
>> C=fliplr(Data);
>> D=flipud(Data);
%多次旋转矩阵
>> E=rot90(Data);
>> F=rot90(E);
>> G=rot90(F);
>> H=rot90(G).
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果：

```
Data =
     1     2     3     4
     5     6     7     8
     9    10    11    12
B =
     1     5     9
     2     6    10
     3     7    11
     4     8    12
C =
     4     3     2     1
     8     7     6     5
    12    11    10     9
D =
     9    10    11    12
     5     6     7     8
     1     2     3     4
E =
     4     8    12
     3     7    11
     2     6    10
     1     5     9
F =
    12    11    10     9
     8     7     6     5
     4     3     2     1
G =
     9     5     1
    10     6     2
    11     7     3
    12     8     4
H =
     1     2     3     4
     5     6     7     8
     9    10    11    12
```

上面的程序代码中演示了各种转换命令，下面简要介绍各种命令的含义。

- fliplr：以数组的垂直中线为对称轴，交换左右对称位置上的数组元素。
- flipud：以数组的水平中线为对称轴，交换左右上下对称位置上的数组元素。
- rot90：逆时针旋转二维数组 90° 。

选取三角矩阵

例 2.10 选取数组上三角或者下二角矩阵。

Step 1 在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> Data=[1,2,3,4;5,6,7,8;9,10,11,12];
>> Al=tril(Data);
>> Bl=tril(Data,1);
>> Cl=tril(Data,2);
>> Dl=tril(Data,-1);
>> Dl=tril(Data,-2);
>> Au=triu(Data);
>>Bu=triu(Data,1);
>>Cu=triu(Data,2);
>>Du=triu(Data,-1);
>>Eu=triu(Data,-2);
```

Step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果：

```
Data =
     1     2     3     4
     5     6     7     8
     9    10    11    12
Al =
     1     0     0     0
     5     6     0     0
     9    10    11     0
Bl =
     1     2     0     0
     5     6     7     0
     9    10    11    12
Cl =
     1     2     3     0
     5     6     7     8
     9    10    11    12
Dl =
     0     0     0     0
     5     0     0     0
     9    10     0     0
El =
     0     0     0     0
     0     0     0     0
     9     0     0     0
Au =
     1     2     3     4
     0     6     7     8
     0     0    11    12
```

```
Bu =
     0     2     3     4
     0     0     7     8
     0     0     0    12
Cu =
     0     0     3     4
     0     0     0     8
     0     0     0     0
Du =
     1     2     3     4
     5     6     7     8
     0    10    11    12
Eu =
     1     2     3     4
     5     6     7     8
     9    10    11    12
```

[illegible]

2.2.4 Kronecker 乘法

例 2.11 演示Kronecker乘法的不可交换性。

step 1 在MATLAB的[illegible]

```
>> X=[1,2;3,4];
>> [illegible]
>> A=kron(X,I);
>> B=kron(I,X);
```

step 2 [illegible]

```
X =
     1     2
     3     4
I =
     1     0     0
     0     1     0
     0     0     1
A =
     1     0     0     2     0     0
     0     1     0     0     2     0
     0     0     1     0     0     2
     3     0     0     4     0     0
     0     3     0     0     4     0
     0     0     3     0     0     4
B =
     1     2     0     0     0     0
     3     4     0     0     0     0
     0     0     1     2     0     0
     0     0     3     4     0     0
     0     0     0     0     1     2
     0     0     0     0     3     4
```

从上面的结果中不难看出，对不同维的两个矩阵，交换它的维时是不同的结果，因此在使用了 rot90(A,k) 命令时不可交换性，不过 rot90(A,k) 命令，请使用户自己去实践和反复比较。

225 高维数组的对称交换

C++高维数组，由于在结构上多了维度，其在操作方法上多了一些操作其他维度的命令，在本节中主要介绍一个简单实例来介绍函数的使用方法。

例 2.12 对三维数组进行对称交换。

step 1 在 MATLAB 的命令窗口中输入下面的程序代码

```
>> data=[1,2,3,4;5,6,7,8;9,10,11,12];
>> A=reshape(data,2,2,3);
>> B=flipdim(A,1);
>> C=flipdim(A,2);
>> D=flipdim(A,3);
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果

```
>> A
A(:,:,1) =
     1     9
     5     2
A(:,:,2) =
     6     3
    10     7
A(:,:,3) =
    11     8
     4    12
>> B
B(:,:,1) =
     5     2
     1     9
B(:,:,2) =
    10     7
     6     3
B(:,:,3) =
     4    12
    11     8
>> C
C(:,:,1) =
     9     1
     2     5
C(:,:,2) =
     3     6
     7    10
C(:,:,3) =
     8    11
    12     4
>> D
D(:,:,1) =
```

```
    11     8
     4    12
D(:,:,2) =
     6     3
    10     7
D(:,:,3) =
     1     9
```

在MATLAB中，shiftdim(A,n)命令中第一个输入参量A表示所要移位的数组，第二个输入参量n指示所要移位的维数。[illegible]

226 高维数组的维序号移动

例2.13 对三维数组进行"维序号移动"。

step 1 在MATLAB的命令窗口中输入下面程序[illegible]

```
>> Data=[1,2,3,4,5,6,7,8,9,10,11,12];
>> A=reshape(Data,2,2,3);
>> Adim=shiftdim(A,1);
>> Adim2=shiftdim(A,2);
>> Adim3=shiftdim(A,3);
```

step 2 查看相关结果。在命令窗口输入变量名称，[illegible]

```
>> A
A(:,:,1)
     1     3
     5     7
A(:,:,2)
     2     4
     6     8
A(:,:,3) =
    11     9
     4    12
>> Adim
Adim(:,:,1) =
     1     6    11
     9     3     8
Adim(:,:,2) =
     5    10     4
     2     7    12
>> Adim2
Adim2(:,:,1) =
     1     5
     6    10
    11     4
Adim2(:,:,2) =
     2     7
     3     7
     8    12
```

```
>> Adim3
Adim3(:,:,1) =
     1     9
     5     2
Adim3(:,:,2) =
     6     3
    10     7
Adim3(:,:,3) =
    11     8
     4    12
```

高维数组的广义共轭转置

例 2.14 对三维数组进行广义共轭转置。

step 1 在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> A=reshape(Data,2,2,3);
>> Ap1=permute(A,[ 1,2,3] );
>> Ap2=permute(A,[ 2,3,1] );
>> Ap3=permute(A,[ 3,2,1] );
>> Ap4=permute(A,[ 3,1,2] );
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果：

```
>> A
A(:,:,1) =
     1     9
     5     2
A(:,:,2) =
     6     3
    10     7
A(:,:,3) =
    11     8
     4    12
>> Ap1
Ap1(:,:,1) =
     1     9
     5     2
Ap1(:,:,2) =
     6     3
    10     7
Ap1(:,:,3) =
    11     8
     4    12
>> Ap2
Ap2(:,:,1) =
     1     6    11
     9     3     8
Ap2(:,:,2) =
     5    10     4
     2     7    12
>> Ap3
Ap3(:,:,1) =
     1     9
```

```
     6     3
    11     6
Ap3(:,:,2) =
     5     2
    10     7
     4    12
>> Ap4
Ap4(:,:,1) =
     1     5
     6    10
    11     4
Ap4(:,:,2) =
     9     2
     3     7
     8    12
```

在MATLAB中，permute命令只接受一个输入参量多维数组数组。第一个参数为向量，其元素是对应维数的排列。[illegible]

228 高维数组的降维操作

例2.15 使用squeeze命令来撤销"孤维"，使高维数组进行降维。

step 1 在MATLAB的命令窗口中输入下面的程序代码

```
>> B=cat(4,A(:,:,1),A(:,:,2),A(:,:,3));
>> C=squeeze(B);
>> size_B=size(B);
>> size_C=size(C);
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果

```
>> B
B(:,:,1,1) =
     1     9
     5     2
B(:,:,1,2) =
     6     3
    10     7
B(:,:,1,3) =
    11     8
     4    12
>> C
C(:,:,1) =
     1     9
     5     2
C(:,:,2) =
     6     3
    10     7
C(:,:,3) =
    11     8
     4    12
>> size_B
```

```
size_B =
     2     2     1     3
>> size_C
size_C =
     2     2     3
```

稀疏矩阵

在MATLAB中，系统一般使用两种方法来存储数据，也就是满矩阵的形式和稀疏矩阵的形式，简称满矩阵和稀疏矩阵。在很多情况下一个矩阵中只有少数元素是非零的，对于满矩阵MATLAB会使用相同的空间来储存零元素和非零元素，这种储存方法对于大多数元素为0的稀疏矩阵而言，将会造成大量的浪费。因此，对于稀疏矩阵，MATLAB提供特殊的存储方法，同时提供特殊的操作函数和运算法则，下面详细介绍。

稀疏矩阵的存储方式

在MATLAB中，一般使用3个矩阵来存储稀疏矩阵，假设有一个 $m \times n$ 的矩阵，其中有 nnz 个非零元素，存储在长度为 nz max 的矩阵中。

- 第一个矩阵用来存储所有的非零元素，该矩阵的长度为 nz max。
- 第二个矩阵用来存储所有的非零元素的行指标，该矩阵的长度也是 nz max。
- 第三个矩阵用来存储每一列的开始处指针和一个标志着这3个矩阵结束的指针，该矩阵的长度为 n+1。

根据上面的介绍，一个稀疏矩阵需要存储 nz max 个浮点数和 nz max+n+1 个整数，因此，存储一个稀疏矩阵需要 8* nz max+4*(nz max+n+1)个字节的单元。

对于稀疏矩阵和满矩阵的存储差异，MATLAB提供下面的转换命令：

- SM = Sparse（A）将其他存储方式转换为其他的稀疏矩阵形式；
- FM = Full（A）把矩阵存储方式从任何一个存储形式转换为满矩阵形式。

2.3.2 使用sparse命令创建稀疏矩阵

由于满矩阵的运算得到的结果还是满矩阵，因此如果不通过相应的命令将不会创建稀疏矩阵。在MATLAB中，提供多个命令来创建稀疏矩阵，经常使用的有sparse和spdiags两种，对应的调用格式如下：

`S = sparse(i,j,s,m,n,nzmax)` 使用[I,j,s] 的行创建 $m \times n$ 维稀疏矩阵S；

- 在上面的命令中，s表示的是按照排列的所有非零元素构成的向量。i，j分别表示非零元素的行下标和列下标向量。

`A = spdiags(B,d,m,n)` 抽取和创建带、对角稀疏矩阵

- 在上面的参数中，m、n分别表示指定矩阵的行和列的维数。d表示的是长度为p的整数向量，B是满矩阵，用来指定A矩阵的对角线位置上的元素。

说明 在MATLAB中，还提供了部分数据稀疏矩阵的测试命令gallery，用户可以直接调用load命令加载外部数据，然后使用上述命令创建稀疏矩阵。

例2.16 使用sparse命令创建稀疏矩阵。

step 1 在MATLAB的命令窗口中输入下面的程序代码。

```
A = [ 0   0   0   5
      1   2   0   0
      1   3   0   0
      0   0   4   0];
S1 = sparse(A);
B=full(S1);
n=5;
D = sparse(1:n,1:n,-2*ones(1,n),n,n);
E = sparse(2:n,1:n-1,ones(1,n-1),n,n);
S2= E+D+E';
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果。

```
S1 =
   (3,1)        1
   (4,2)        2
   (3,2)        3
   (4,3)        4
   (1,4)        5
B =
     0     0     0     5
     0     2     0     0
     1     3     0     0
     0     0     4     0
D =
   (1,1)       -2
   (2,2)       -2
   (3,3)       -2
   (4,4)       -2
   (5,5)       -2
E =
   (2,1)        1
   (3,2)        1
   (4,3)        1
   (5,4)        1
S2 =
   (1,1)       -2
   (2,1)        1
   (1,2)        1
   (2,2)       -2
   (3,2)        1
   (2,3)        1
   (4,3)        1
   (3,4)        1
   (4,4)        2
   (5,4)        1
   (4,5)        1
```

```
   (5,5)        -2
```

step 3 查看存储信息。为了加强用户对稀疏矩阵的存储信息的理解，可以使用whos命令查看各变量的信息：

```
>> whos
  Name      Size                    Bytes  Class
  A         4x4                       128  double array
  B         4x4                       128  double array
  D         5x5                        84  double array (sparse)
  E         5x5                        72  double array (sparse)
  S1        4x4                        80  double array (sparse)
  S2        5x5                       180  double array (sparse)
  n         1x1                         8  double array
Grand total is 60 elements using 680 bytes
```

使用 spdiags 命令创建稀疏矩阵

例 2.17 使用 spdiags 命令来创建稀疏矩阵。

step 3 在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> B = [ 41    11     0
         52    22     0
         63    33    13
         74    44    24 ];
>> d = [ -3
         0
         2] ;
>> S=spdiags(B,d,7,4);
>> D=full(S);
```

step 3 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果：

```
S =
   (1,1)        11
   (4,1)        41
   (2,2)        22
   (5,2)        52
   (1,3)        13
   (3,3)        33
   (6,3)        63
   (2,4)        24
   (4,4)        44
   (7,4)        74
D =
    11     0    13     0
     0    22     0    24
     0     0    33     0
    41     0     0    44
     0    52     0     0
     0     0    63     0
     0     0     0    74
```

[illegible]

2.3.4 查看稀疏矩阵的信息

[illegible]

- n=nnz(x) 查看稀疏矩阵中的非零元素个数。
- s=nonzeros(A) 返回稀疏矩阵中的非零元素数值。
- n=nzmax(S) 返回稀疏矩阵中存储非零元素的空间长度。

例2.18 查看某稀疏矩阵的元素信息。

step 1 在MATLAB命令窗口中输入如下程序代码：

```
>> load west0479
>> S=west0479;
>> n1=nnz(S);
>> s1=nonzeros(S);
>> n2=nzmax(S);
>> format short e
```

step 2 查看程序结果，在命令窗口输入变量名称，可以得到下面的程序运行结果：

```
n1 =
        1987
n2 =
        1887
s1 =
 1.0000e+000
-3.7648e-002
-3.4424e-001
 1.0000e+000
-2.4523e-002
-3.7371e-001
[illegible]
-3.6613e-002
[illegible]//省略了部分数据
 3.6044e-001
 2.0539e-002
 7.1093e-001
 3.1305e+000
 1.3574e-001
 1.0000e+000
 2.8831e-001
[illegible]
```

[illegible]

235 稀疏矩阵的图形化信息

除了上述的信息查询函数之外，MATLAB提供查看稀疏矩阵的图形化命令spy，其具体的调用格式为

spy(S,'LineSpec',markersize)，其中S表示的是稀疏矩阵，LineSpec表示的是线型属性的字符串，markersize则表示标记大小。

例2.19 查看稀疏矩阵的图形化信息。

step 1 在MATLAB的命令窗口中输入下面的程序代码：

```
>> load west0479
>>S=west0479;
>> spy(S)
```

step 2 在输入上面的程序代码后，按"Enter"键，得到的图形如图2.2所示。

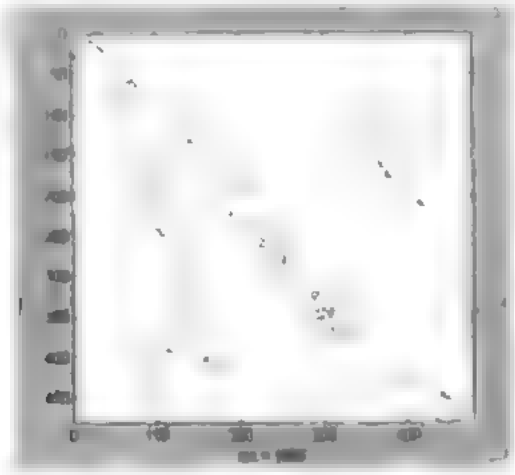


图2.2 稀疏矩阵S的图形

step 3 在MATLAB的命令窗口中输入下面的程序代码

```
>> B=bucky;
>> C=B^2;
>> D=B^4;
>> E=B^6;
>>subplot(221);
>>spy(B)
>>subplot(222);
>>spy(C)
>>subplot(223);
>>spy(D)
>>subplot(224);
>>spy(E)
```

step 4 在输入上面的程序代码后，按"Enter"键，得到的图形如图2.3所示。

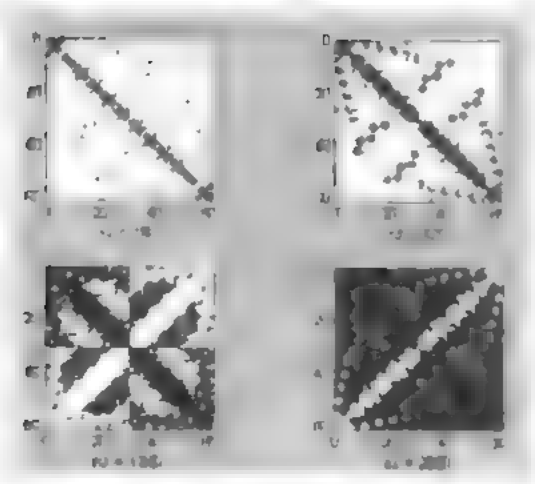


图2.3 稀疏矩阵的图形

2.4 字符串数组

[illegible]

2.4.1 直接输入法创建字符串数组

[illegible]

例2.20 通过直接输入法来创建字符串数组。

step 1 [illegible]

```
[illegible]
b='字符串数组';
[illegible]
[illegible]
```

step 2 [illegible]

```
a =
Char
b =
字符串数组
[illegible]
Example'2.1'
d =
字符串数组 Example'2.1'.
```

从上面的程序代码可以得知创建字符串数组的基本方法

- 直接在单引号对内输入字符串的内容。

- [illegible]，每个单引号字符可需要使用连续的2个单引号字符
- 可以使用小的字符串构成大的字符串。

2.4.2 使用ASCII码创建字符串数组

例2.21 通过ASCII码的转换来创建字符串数组。

step 1 在MATLAB的命令窗口中输入下面的程序代码

```
>> b='字符串数组';
>> ascii_b=double(b);
>> c=char(ascii_b);
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果

```
b =
字符串数组
ascii_b =
       23383       31526       20018       25968       32452
c =
字符串数组
```

从上面的结果可以看出，在MATLAB中，用户可以使用double和char在字符串数组和数值数组之间转换，其中数值数组是字符串中的ASCII码

step 3 通过ASCII码实现字符串变量大小写的转换。在命令窗口中输入下面的程序代码

```
>> charA='Matlab 7.0 String Data';
>> w=find(charA>='a'&charA<='z');
>> asciiA=double(charA);
>> asciiA(w)=asciiA(w)-32;
>> charB=char(asciiA);
```

step 4 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果

```
charA =
Matlab 7.0 String Data
charB =
MATLAB 7.0 STRING DATA
```

在MATLAB中，大小写字符相差的ASCII码的相差值为32，因此使用数值数组的一个数加32或减32就实现大小写互换。

2.4.3 使用函数创建字符串数组

例2.22 通过数组构建函数来创建字符串数组。

step 1 在MATLAB的命令窗口中输入下面的程序代码

```
>> ch1=char('Matlab 7.0','String Data');
```

```
>> ch2=str2mat('Matlab','7.0','','String','Data');
>> t1 = 'first';
>> t2 = 'string';
>> t3= 'matrix';
>> t4 = 'second';
>> S1 = strvcat(t1, t2, t3);
>> S2 = strvcat(t4, t2, t3);
>> S3 = strvcat(S1, S2);
```

step 2 查看程序结果，在命令窗口输入查看名称，[illegible]

```
ch1 =
Matlab 7.0
String Data

ch2 =
Matlab
7.0

String
Data

S1 =
first
string
matrix

S2 =

second
string
matrix

S3 =

first
string
Matrix
second
string
matrix
```

在MATLAB中，使用字符串数组来处理字符串时，不管是字符串的长度不同，[illegible]

2.4.4 处理字符串数组的空格

为了方便用户操作字符串数组，MATLAB中提供多种字符串的操作函数，[illegible]字符串进行转换、[illegible]、查找等功能，这些函数内容繁多，在本章中将[illegible]字符串操作函数，其他具体函数请用户自行查阅帮助文件。

例2.23 使用不同的方法来处理字符串中的空格。

step 1 在MATLAB的命令窗口中输入下面程序代码：

```
>> A(1,1) = 'MATLAB     ';
>> A(1,2) = 'SIMULINK     ';
>> A(2,1) = 'Toolboxes     ';
>> A(2,2) = 'The MathWorks     ';
>> B=deblank(A);
>> cstr = ['     Trim leading white-space';
'Trim trailing white-space     '];
>>cstrim=strtrim(cstr);
```

Step 2 在命令行窗口中，输入变量名称，按回车键，即可查看变量的内容。

```
A =
    'MATLAB     '    'SIMULINK     '
    'Toolboxes     '    'The MathWorks     '
B =
    'MATLAB'    'SIMULINK'
    'Toolboxes'    'The MathWorks'
cstr =
    '     Trim leading white-space'
    'Trim trailing white-space     '
cstrim =
    'Trim leading white-space'
    'Trim trailing white-space'
```

从上面的结果可以看出，函数deblank()只能删除字符串末尾的空格，但是其结构数组中的字符串长度不变，strtrim()命令的功能，是删除字符串前面和末尾的空格，deblank()命令的功能则是删除字符串末尾的空格。

2.4.5 读取字符串数组的信息

例2.24 使用函数来查找、替代和读取字符串中的信息。

Step 1 创建字符串，在MATLAB命令窗口，输入下面的命令。

```
>> s1 = 'This is a good example.';
>>str = strrep(s1, 'good', 'great');
```

Step 2 查看变量的内容，在命令窗口输入变量名称，按回车键，即可查看变量的内容。

```
s1 =
This is a good example.
str =
This is a great example.
```

在上面的例子中，字符串变量s1中的字符串"good"被替换为"great"。

Step 3 创建一个字符串的数组，在MATLAB命令窗口，输入下面的命令。

```
>> str = '<table border=5 width="100%" cellspacing=0>';
```

```
[border width space] = strread(str, ...
    '%*s%*s %c %*s "%4s" %*s %c', 'delimiter', '= ');
```

Step 4 查看程序结果，按"Enter"键，[illegible]面的程序结果

```
border =
=
width =
    [illegible]
space =
0
```

在上述程序代码中，调用函数 strread 分别为变量 border、width 和 space 赋值。

Step 5 删除了符号后的"="符号，得出有效信息。在 MATLAB 的命令窗口中输入一个字符串变量 str

```
>> s = sprintf('%s%s%s',  ...
'<ul class=continued><li class=continued><pre>', ...
'<a name="13474"></a>token = strtok(''str'', delimiter)',  ...
'<a name="13475"></a>token = strtok(''str'')');
rem = s;
for k=1:11
    [t(k), rem] = strtok(rem, '<>');
    if isempty(t(k)),  break;  end
    disp(sprintf('%s',t(k)));
    end
```

Step 6 查看程序结果。按"Enter"键，可以得到下面的程序结果

```
ul class=continued
li class=continued
pre
a name="13474"
/a
token = strtok('str', delimiter)
a name="13475"
/a
token = strtok('str')
```

在上述程序代码中，首先将字符串变量 s 的内容赋值给变量 rem，然后将 rem 中的字符串以"<"和">"为分隔符，分别赋值给变量 t 的各个元素，最后使用 disp 函数将 t 变量的各个字符串内容显示出来。

2.5 构架数组

在 MATLAB 中，构架数组是一个比较特殊的数组类型，在该数组中，以字段名称作为索引，某几个数据构架数组的基本单元是结构，数组中的每个构架都是平等的，[illegible]

分"域"之后才能使用，数据不能直接存放在构架数组中，而只能直接存放在构架的域中。而构架中的"域"可以存放任何类型、任何大小的数组，而且不同构架数组中的同名域中可以存放不同的内容。

鉴于构架数组具有上面这么多的特殊之处，本节将介绍如何在MATLAB中创建、操作和运用构架数组。

使用直接法创建单构架数组

构架数组实质上具有面向对象中的数据结构功能，具有属性名称、属性数值两个主要对象域，对此MATLAB提供多种创建方法来构建构架数组。在本小节中，将以具体的实例来说明如何创建构架数组。

例 2.25 使用直接法来创建某个关于病人的构架数组。

step 1 创建维度为 1 × 1 的构架数组。在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> patient.name = 'John Doe';
>>patient.billing = 127.00;
>>patient.test = [ 79 75 73; 180 178 177.5; 220 210 205];
>> patient.medi_information.city='NewYork';
>> patient.medi_information.date='2006/7/1';
```

step 2 查看该构架数组的存储信息。使用 whos 命令查看数组信息，其具体的信息如下：

```
>> whos patient
  Name          Size                    Bytes  Class
  patient       1x1                       870  struct array

Grand total is 39 elements using 870 bytes
```

step 2 查看该构架数组的具体信息。在 MATLAB 的命令窗口中输入下面的代码：

```
>> patient
patient =
                name: 'John Doe'
             billing: 127
                test: [ 3x3 double]
    medi information: [ 1x1 struct]

>> patient.medi_information
ans =
    city: 'NewYork'
    date: '2006/7/1'
>> patient.test
ans =
   79.0000   75.0000   73.0000
  180.0000  178.0000  177.5000
  220.0000  210.0000  205.0000
>> patient.medi_information.city
ans =
NewYork
```

在上面的程序代码中，patient 只有一个构架，其总共有 4 个数值域：name、billing、test 和

medi_information，其中medi_information，在city和date两个字段，分别直接赋值给构架数组的最简单方法。

[illegible]

2.5.2 使用直接法创建二维构架数组

例2.26 沿用上面的实例，创建二维构架数组。

step 1 直接扩展构架数组的维度。在MATLAB主命令窗口中输入下面的语句：

```
>>patient(2).name = 'Ann Lane';
>>patient(2).billing = 28.50;
>>patient(2).test = [68 70 68; 118 118 119; 172 170 169];
>>patient(2).medi_information.city='Chicago';
>>patient(2).medi_information.date='2006/8/1';
>>patient(3).name = 'Ann Smitch';
>>patient(3).billing = 504.70;
>>patient(3).test = [80 80 68; 153 153 156; 181 190 182];
>>patient(3).medi_information.city='Boston';
[illegible]
```

step 2 查看构架数组所存储信息。使用whos命令查看数组信息，其结果信息为：

```
>> whos patient
  Name          Size                    Bytes  Class
  patient       1x3                      2100  struct array

Grand total is 118 elements using 2100 bytes
```

step 3 [illegible]

```
[illegible]
```

step 4 [illegible]

```
>> mypatient
mypatient =

1x2 struct array with fields:
    name
    billing
    test
    medi_information
```

step 5 [illegible]

```
>> mypatient(1)
ans =
```

```
                    [illegible]
                 billing: 127
                    test: [3x3 double]
    medi_information: [1x1 struct]
>> name=mypatient(2).name
name =
Ann Lane
>> test=mypatient(2).test
test =

    68    70    68
   118   118   119
   172   170   169
```

[illegible]

2.5.3 使用直接法创建三维构架数组

例 2.27 沿用上面的实例，创建三维构架数组。

step 1 [illegible]在 MATLAB 的命令窗口中输入下面的语句：

```
patient(1,1,1).name = 'John Doe';patient(1,1,1).billing = 127.00;
[illegible] = [79 75 73; 180 178 177.5; 220 210 205];
patient(1,2,1).name = 'Ann Lane';patient(1,2,1).billing = 28.50;
[illegible] = [68 70 68; 118 118 119; 172 170 169];
[illegible] = 'Al Smith';patient(1,1,2).billing = 504.70;
patient(1,1,2).test = [80 80 80; 153 153 154; 181 190 182];
[illegible] = 'Dora Jones';patient(1,2,2).billing = 1173.90;
patient(1,2,2).test = [73 73 75; 103 103 102; 201 198 200].
```

step 2 [illegible]使用 who 命令查看数组信息，其具体的结构如下：

```
>> patient
[illegible]

1x3x2 struct array with fields:
    name
    billing
    [illegible]
    [illegible]
```

step 3 [illegible]，如图 2.4 所示。

2.5.4 使用命令创建构架数组

例 2.28 使用 struct 命令创建构架数组。

图2.4 三维构架数组的图形化结构

Step 1 [illegible]

```
>>W = struct('city', {'Bos','Chi','Lin','Dnv','Vgs','SFr'}, ...
             'temp',  [illegible], ...
             'heatix', [illegible], ...
             'wspeed', [8, 3, 11, 9, 4, 18], ...
             [illegible]
```

Step 2 [illegible]

```
>> W
W =

1x6 struct array with fields:
    city
    temp
    heatix
    wspeed
    [illegible]
```

[illegible]

2.5.5 访问构架数组的数据

[illegible]

例2.29 访问构架数组的数据。

Step 1 [illegible]

```
>> my1=mypatient(1);
>> test2b = patient(3).test(2,2);
>> bills = [patient.billing];
>> tests = [patient(1:2).test];
```

step 2 在程序运行中，在命令窗口输入变量名称，可以得到上面的程序运行结果

```
>> my1
my1 =

                name: 'John Doe'
            birthdy: 127
               test: [3x3 double]
    medi_information: [1x1 struct]

test2b =
   163

bills =
  127.0000   28.5000  504.7000
tests =

    [3x3 double]    [3x3 double]
```

在上面的程序代码中，[illegible] 变量是前面已经定义的结构数组，而后的变量则是不完整的引用方法

例 2.30 在程序运行过程中调用结构数组中的数据，并绘制图形表示

step 1 创建结构数组。在 MATLAB 的命令窗口中输入下面的程序代码

```
test(1).lead = .007;
test(2).lead = .031;
test(3).lead = .019;
test(1).mercury = .0021;
test(2).mercury = .0009;
test(3).mercury = .0013;
test(1).chromium = .025;
test(2).chromium = .017;
test(3).chromium = .1
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到上面的程序运行结果

```
test =

1x3 struct array with fields:
    lead
    mercury
    chromium
```

step 3 编写绘图函数命令。选择命令窗口编辑菜单下的 "New" → "M-file" 命令，打开 M 文件编辑器，在其中输入下面的程序代码

```
function [r1, r2] = concen(toxtest);
% 计算变量之间的相对比例
r1 = [toxtest.mercury] ./ [toxtest.lead];
r2 = [toxtest.lead] ./ [toxtest.chromium];
%计算变量数值
```

```
lead = [toxtest.lead];
mercury = [toxtest.mercury];
chromium = [toxtest.chromium];
%绘制三个变量的图形
plot(lead, 'r','LineWidth',1.5);   hold on
plot(mercury, 'b','LineWidth',1.5)
[illegible]
%添加标题图例和标题
legend('lead','mercury','chromium')
title('The ratio of mix')
grid
```

[illegible]

step 4 [illegible] 在 MATLAB 命令行窗口中输入 [illegible]

```
>> [a,b]=concen(test)
```

step 5 查看结果，输入上面命令后运行结果，[illegible]

```
a =
    0.3000    0.0290    0.0684
b =
    0.2800    1.8235    0.1900
```

同时，可以得到的图形如图 2.5 所示。

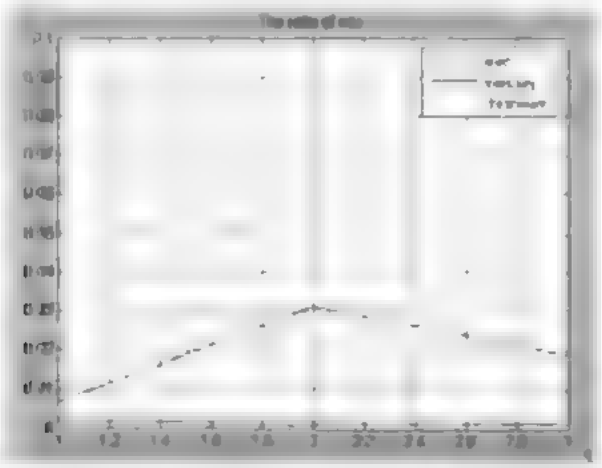


图 2.5 绘制结果的图形

2.5.6 设置构架数组的域属性

由于构架数组的域是存放数据的场所，因此选取和设置构架数组中数据的前提就是了解或者熟悉构架数组的域名。只要在前面两节介绍了访问构架数组域名的方法，但是这种方法并不能获得其他命令能够处理的域名。为了解决这个矛盾，MATLAB 提供了处理域名的各种命令，在本小节中将主要介绍一些常见命令。

- names = fieldnames(s)：获取构架数组的域名
- f = getfield(s,'field')：获取具体构架数组中的内容

✦ s = setfield(s, 'field', v) 设置具体构架数组中的内容

例 2.31 使用 fieldnames 命令获取构架数组的域属性。

step 1 创建构架数组。在 MATLAB 的命令窗口中输入下面的程序代码：

```
mystr(1,1).name = 'alice';
mystr(1,1).ID = 0;
mystr(2,1).name = 'gertrude';
mystr(2,1).ID = 1
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果：

```
mystr =
2x1 struct array with fields:
    name
    ID
```

step 3 获取域名属性。在 MATLAB 的命令窗口中输入下面的程序代码：

```
n = fieldnames(mystr)
```

step 4 查看程序结果。按"Enter"键，可以得到下面的程序结果：

```
n =
    'name'
    'ID'
```

step 5 获取其他对象的域名属性。在 MATLAB 的命令窗口中输入下面的程序代码：

```
f = java.awt.Frame;
fieldnames(f)
```

step 6 查看程序结果。按"Enter"键，可以得到下面的程序结果：

```
ans =
    'DEFAULT_CURSOR'
    'CROSSHAIR_CURSOR'
    'TEXT_CURSOR'
    'WAIT_CURSOR'
    ..................................//省略了部分数据
    'BOTTOM_ALIGNMENT'
    'LEFT_ALIGNMENT'
    'RIGHT_ALIGNMENT'
    'WIDTH'
    'HEIGHT'
    'PROPERTIES'
    'SOMEBITS'
    'FRAMEBITS'
    'ALLBITS'
    'ERROR'
    'ABORT'
```

例 2.32 使用 getfield 命令来获取域属性。

step 1 创建构架数组。在 MATLAB 的命令窗口中输入下面的程序代码：

```
mystr(1,1).name = 'alice';
mystr(1,1).ID = 0;
mystr(2,1).name = 'gertrude';
mystr(2,1).ID = 1
```

step 2 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果：

```
mystr =
2x1 struct array with fields:
    name
    ID
```

step 3 获取域名属性。在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> for k = 1:2
    name{k} = getfield(mystr,{k,1},'name');
end
```

step 4 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果：

```
name =

    'alice'    'gertrude'
```

例 2.33 使用 setfield 命令来设置域属性。

step 1 创建构架数组。在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> class = 5;   student = 'John_Doe';
grades_Doe = [85, 89, 76, 93, 85, 91, 68, 84, 95, 73];
grades = [];
grades = setfield(grades, {class}, student, 'Math', ...
   {10, 21:30}, grades_Doe);
```

step 4 查看程序结果。在命令窗口输入变量名称，可以得到下面的程序结果：

```
grades =

1x5 struct array with fields:
    John_Doe
```

step 3 查看构架数组的信息。在 MATLAB 的命令窗口中输入下面的程序代码：

```
grades(class).John_Doe.Math(10, 21:30)
```

step 4 查看程序结果。按"Enter"键，可以得到下面的程序结果：

```
ans =

85    89    76    93    85    91    68    84    95    73
```

小结

在本章中，依次介绍了如何在MATLAB中创建和操作数值数组、稀疏矩阵、字符串数组和构架数组。这些内容都是MATLAB中的基础知识，因此希望用户能够熟练掌握。其中数值数组是MATLAB所有操作的重要内容，字符串和构架数组在实际工程中也有广泛的应用。在后面的章节中，将主要介绍如何使用MATLAB进行数值运算。

✦ ✦ ✦

第3章 数值运算

本章包括

- 矩阵分析
- 矩阵分解
- 概率论和数理分析
- 线性方程组
- 数值积分

在前面已经介绍过，MATLAB的基本运算单元是数组，因此本章将从矩阵分析、线性代数的数值计算开始介绍，然后介绍函数的零点、数值积分、数理统计和分析等。在MATLAB中，数值运算主要通过函数或者命令来实现。由于在MATLAB中所有的数据都以矩阵的形式出现，其对应的数值运算包括两种类型：一种是针对整个矩阵的数值运算，也就是矩阵运算，例如求解矩阵行列式的函数det；另外一种是针对矩阵中的元素进行运算的函数，可以称为矩阵元素的运算，例如求解矩阵中每个元素的余弦函数cos等。

从整体的角度来看，本章的内容之间没有太大的递承关系，用户可以根据需要来选择阅读的章节。同时，为了方便大家阅读各自章节的内容，每个小节中的例子都是完整的，用户可以在自己的机器上运行。

矩阵分析

矩阵分析是线性代数的重要内容，也是几乎所有MATLAB函数的分析的基础。在MATLAB7.0中，可以支持多种线性代数中定义的操作，正是其强大的矩阵运算能力才使得MATLAB成为优秀的数值计算软件。本节将主要介绍关于矩阵分析的内容。

使用norm函数进行范数分析

根据线性代数的知识，对于线性空间中的某个向量$x=\{x_1, x_2, \cdots, x_n\}$，其对应的$P$级范数的定义为$\|x\|_p=(\sum_{i=1}^{n}|x_i|^p)^{1/p}$，其中的参数$p=1, 2, \cdots, n$。同时，为了保证整个定义的完整性，定义范数数值$\|x\|_\infty=\max_{1\le i\le n}|x_i|$，$\|x\|_{-\infty}=\min_{1\le i\le n}|x_i|$。

矩阵范数的定义是基于向量的范数而定义的，具体的表达式为：

$$\|A\|=\max_{\forall x\neq 0}\frac{\|Ax\|}{\|x\|}$$

在实际应用中，比较常用的矩阵范数是1、2和∞阶范数，其对应的定义如下：

$$\|A\|_1=\max_{1\le j\le n}\sum_{i=1}^{n}|a_{ij}|,\ \|A\|_2=\sqrt{S_{\max}\{A^TA\}}\text{和}\|A\|_\infty=\max_{1\le i\le n}\sum_{j=1}^{n}|a_{ij}|$$

在上面的定义式$\|A\|_2=\sqrt{S_{\max}\{A^TA\}}$中，$S_{\max}\{A^TA\}$表示矩阵$A$的最大奇异值的平方，关于奇异值的定义将在后面章节中介绍。

[illegible]

在 MATLAB 中，求解向量和矩阵范数的命令如下

- n = norm(A)　计算向量或者矩阵的 2 阶范数
- n = norm(A,p)　计算向量或者矩阵的 p 阶范数

[illegible]

例 3.1　根据定义和 norm 来分别求解向量的范数。

step 1 进行范数求解。选择命令窗口编辑栏中的"File"→"New"→"M-file"命令，打开 M 文件编辑器，在其中输入下面的程序代码

```
%输入向量
x=[1:6];
y=x.^2;
%使用定义求解各阶范数
N2=sqrt(sum(y));
Ninf=max(abs(x));
Nvinf=min(abs(x));
%使用norm命令求解范数
n2=norm(x);
ninf=norm(x,inf);
nvinf=norm(x, -inf);
%输出求解的结果
disp('The method of definition:')
fprintf('The 2-norm is %6.4f\n',N2)
fprintf('The inf-norm is %6.4f\n',Ninf)
fprintf('The minusinf-norm is %6.4f\n',Nvinf)
fprintf('\n//----------------------------//\n\n')
disp('The method of norm command:')
fprintf('The 2-norm is %6.4f\n',n2)
fprintf('The inf-norm is %6.4f\n',ninf)
fprintf('The minusinf-norm is %6.4f\n',nvinf)
```

在输入上面的代码后，将该程序代码保存为"normex.m"文件。

step 2 查看计算结果。在 MATLAB 的命令窗口中输入"normex"后，按"Enter"键，得到计算结果如下

```
The method of definition:
The 2-norm is 9.5394
The inf-norm is 6.0000
The minusinf-norm is 1.0000

//----------------------------//

The method of norm command:
The 2-norm is 9.5394
```

```
The inf-norm is 6.0000
The minusinf-norm is 1.0000
```

[illegible]

例3.2 根据定义和norm来分别求解Hilbert矩阵的范数。

Step 1 进入[illegible]，选择命令窗口，编辑"文件"→"New"→"M-file"命令，打开M文件编辑器，在其中输入下面的程序代码

```
%输入矩阵
A=hilb(5);
%使用定义求解各阶范数
N1=max(sum(abs(A)));
N2=norm(A);
N[illegible]=max([illegible](A'))[illegible];
N[illegible]=sqrt(max(eig(A'*A)));
%使用norm命令求解范数
n1=norm(A,1);
n2=norm(A,2);
ninf=norm(A,inf);
nfro=norm(A,'fro');
%输出求解的结果
disp('The method of definition:')
fprintf('The 1-norm is %6.4f\n',N1)
fprintf('The 2-norm is %6.4f\n',N2)
fprintf('The [illegible] is %6.4f\n',N[illegible])
fprintf('The Ff-norm is %6.4f\n',N[illegible])
fprintf('\n//-------------------------------//\n\n')
disp('The method of norm command:')
fprintf('The 1-norm is %6.4f\n',n1)
fprintf('The [illegible] %6.4f\n',n2)
fprintf('The inf-norm is %6.4f\n',ninf)
fprintf('The F-norm is %6.4f\n',nfro)
```

在输入上面的代码后，将该程序代码保存为"normex2.m"文件。

Step 2 为查看计算结果，在MATLAB的命令窗口中输入"normex2"，然后按"Enter"键，得到计算结果如下

```
The method of definition:
The 1-norm is 2.2833
The 2-norm is 1.5671
The inf-norm is 2.2833
The Ff-norm is 1.5809

//-------------------------------//

The method of norm command:
The 1-norm is 2.2833
The 2-norm is 1.5671
The inf-norm is 2.2833
The F-norm is 1.5809
```

step 3 各矩阵的 A 个数值一样，在上述运算中，可以使用norm命令来求解矩阵的范数。矩阵 A 的范数，用户可以查看 A 的具体元素

```
%定义数值显示格式
>> format rat
>> A
A =
     1          1/2        1/3        1/4        1/5
    1/2         1/3        1/4        1/5        1/6
    1/3         1/4        1/5        1/6        1/7
    1/4         1/5        1/6        1/7        1/8
    1/5         1/6        1/7        1/8        1/9
```

在 MATLAB 中，Hilbert 函数是特殊的矩阵函数，主要用来以指定的阶数生成一个二维的希尔伯特矩阵。命令 hilb(n) 命令来生成希尔伯特矩阵，其元素满足等式 A(i,j)=1/(i+j-1)。

3.1.2 使用 normest 函数进行范数分析

当要求分析的矩阵比较大时，求解矩阵范数的时间会比较长，因此当允许某个误差的范数满足某种条件时，可以使用 normest 函数来求解范数。在 MATLAB 帮助文档中，normest 函数主要是用来处理稀疏矩阵的，但是该命令也可以接受非稀疏矩阵的输入，一般用来处理维数比较大的矩阵。

normest 函数的主要调用格式如下：

- nrm = normest(S) 估计矩阵 S 的 2 阶范数数值，默认的允许误差数值增为 1e-6
- nrm = normest(S,tol) 使用参数 tol 作为允许的相对误差。

例 3.3 分别使用 norm 和 normest 命令来求解矩阵的范数。

step 1 在 MATLAB 的命令窗口中输入下面的命令

```
>> W = gallery('wilkinson',500) ;
t1=clock;
W_norm=norm(W);
t2=clock;
t_norm=etime(t2,t1);
t3=clock;
W_normest=normest(W);
t4=clock;
t_normest=etime(t4,t3);
```

在上面的程序代码中，首先使用 Wilkinson 函数生成一个矩阵，然后分别使用 norm 和 normest 命令来求解该矩阵的范数，并使用 tic-toc 命令来计算时间。

step 2 分别计算出矩阵范数。在命令窗口中输入对应的变量名称，得到的结果如下

```
W_norm =
  250.2462
t_norm =
    0.7410
```

```
W_normest =
  250.2168
t_normest =
0.9210
```

[illegible] normest 命令明显要比 norm 命令

step 3 修改矩阵的维度，重新求解范数。在 MATLAB 中输入下面的命令

```
%重新设置矩阵的维度
W = gallery('wilkinson',1000) ;
t1=clock;
[illegible]
[illegible]
t_norm=etime(t2,t1)
[illegible]
b_normest=normest(W)
[illegible]
[illegible]
%显示结果
W_norm =
  500.2462
t_norm =
    8.4620
W_normest =
  500.2116
t_normest =
    4.4270
```

[illegible] 矩阵的范数时，使用 normest 命令。

关于每个命令的计算时间，[illegible]

3.13 条件数分析

在[illegible]条件数，描述线性方程 $Ax=b$ 的解对 b 中[illegible]条件数，其对应的数学定义是

$$k = |A^{-1}||A|$$

[illegible]的条件数为 ∞。而病态矩阵的条件数则比较大。

[illegible]条件数，[illegible]

$$\frac{1}{k}\left(\frac{\delta b}{b}\right) \le \frac{\delta x}{|x|} \le k\left(\frac{\delta b}{b}\right)$$

在 MATLAB 中，求取矩阵 X 的条件数的命令如下

c = cond(X) 求矩阵x的条件数

例3.4 [illegible]，使用矩阵的条件数来对解[illegible]个数解的精度。

step 1 进行数值求解，在MATLAB主窗口中输入下面全部命令

```
>> M=magic(5);
>> b=ones(5,1);
%利用左除来求解近似解
>> x=M\b;
%利用精确的求解
>> xinv=inv(M)*b;
%计算实际相对误差
>> ndb=norm(M*x-b);
>> nb=norm(b);
>> ndx=norm(x-xinv);
>> nx=norm(x);
>> er=ndx/nx;
>> k=cond(M);
%计算最大可能的近似相对误差
>> erk1=k*eps;
%计算最大可能的相对误差
>> erk2=k*ndb/nb;
```

[illegible]

step 2 查看求解结果，在命令窗口中输入计算机变量名称，回车后结果为

```
>> k
k =
    5.4619
>> er
er =
  2.9403e-016
>> erk1
erk1 =
  1.2128e-015
>> erk2
erk2 =
  6.6426e-016
```

[illegible]

step 3 修改求解程序，重新计算求解精度，在命令窗口中输入下面语句

```
M=hilb(12);
[illegible]
x=[illegible];
[illegible]
ndb=norm(M*x-b);
```

```
[illegible]
ndx=norm(x-xinv);
nx=norm(x);
[illegible]
k=cond(M);
erkl=k*eps;
[illegible]
```

step 5 查看本例的结果，在命令窗口中输入计算机变量名称，回车后的结果为

```
>> k
k =
   [illegible]e+01
>> er
[illegible]
    0.0119
>> erkl
erkl =
    3.9846
>> erk2
erk2 =
  5.3479e+007
```

为上面的完整[illegible]的矩阵为病态矩阵，[illegible]是高度的病态，这样会造成比较大的计算误差。

3.1.4 数值矩阵的行列式

在MATLAB中，求解矩阵行列式的命令比较简单，其调用格式如下：

- d = det(A)　　求解矩阵A的行列式，就要输入的参数A不是矩阵，而是一个标量，也就是命令返回的是一个数。

例3.5　求解矩阵的行列式。

step 1 求解矩阵的行列式。在MATLAB的命令窗口中输入下面的命令

```
[illegible]
A=[illegible];
a(i)=det(A);
disp('The Matrix is:');
[illegible]
[illegible]
[illegible]
end
```

step 2 查看本例的结果，在输入[illegible]函数程序代码后，按"Enter"键，则命令行窗口[illegible]

```
The Matrix is:
     1     0     0
     0     0     1
     0     0     0
```

```
The determinant is
2
The Matrix is:
     0     1     0     0
     1     1     0     1
     0     0     1     0
     1     0     1     0
The determinant is
1
The Matrix is:
     0     0     1     1     1
     [illegible]  1     0     0     1
     0     1     1     1     0
     1     0     1     1     1
           1     1     0     1
The determinant is
-2
```

在上面的程序代码中，首先使用rand()函数产生随机矩阵，并在使用det()命令来计算矩阵的行列式。

3.1.5 符号矩阵的行列式

[illegible]，除了数值矩阵的行列式之外，还可以计算符号矩阵的行列式，下面举例说明。

例3.6 求解符号矩阵的行列式。

Step 1 在MATLAB的命令窗口中输入下面的命令

```
[illegible]
[illegible]
>> B=det(A);
[illegible]
```

在上面的程序代码中，使用det()命令计算矩阵的行列式，并将结果赋给B。[illegible]

Step 2 [illegible]

```
A =
 [   sin(t),  cos(t)]
[  -cos(t),  sin(t)]
B =
[illegible]
C =
[illegible]
```

从上面的结果可以看出，使用det()命令也可以求解符号矩阵的行列式。关于符号运算的详细命令，请参阅一些有关符号运算的章节。

矩阵的化零矩阵

对于非满秩的矩阵 A，存在某矩阵 Z，满足 $A \cdot Z=0$，同时矩阵 Z 是一个正交矩阵，也就是说 $Z^T \cdot Z=I$，则矩阵 Z 被称为矩阵 A 的化零矩阵。在 MATLAB 中，求解化零矩阵的命令为 null，其具体的调用格式如下：

- Z = null(A) 返回矩阵 A 的化零矩阵，如果化零矩阵不存在则返回空矩阵；
- Z = null(A,'r') 返回有理数形式的化零矩阵。

例 3.7 求解非满秩矩阵 A 的化零矩阵。

step 1 在 MATLAB 的命令窗口中输入下面的命令：

```
>> A = [1     2     3
     4     5     6
     7     8     9];
>> Z = null(A);
R=A*Z;
```

step 2 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下：

```
>> Z
Z =
    0.4082
    0.8165
   -0.4082
>> R
R =
  1.0e-015 *
         0
   -0.4441
         0
```

step 2 求解有理数形式的化零矩阵。在 MATLAB 的命令窗口中输入下面的命令：

```
>> ZR=null(A,'r');
>> RZ=A*ZR;
```

step 1 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下

```
>> ZR
ZR =
     1
    -2
     1
>> RZ
RZ =
     0
     0
     0
```

线性方程组

线性方程组是线性代数中的主要内容之一，也是理论发展最为完整的部分。在 MATLAB 中也包含

[illegible]，[illegible]。[illegible]两种：恰定方程组、超定方程组和欠定方程组，下面详细介绍。

3.2.1 非奇异线性方程组

[illegible] *A* [illegible] *B* [illegible]，MATLAB 提供 [illegible] *A* [illegible]

- [illegible]精确解
- [illegible]解 NaN

下面分别用例子来详细说明如何使用该命令求解方程组。

例 3.8 求解非奇异矩阵的线性方程的解。

step 1 在 MATLAB 的命令窗口中输入下面的命令

```
>> A = pascal(4);
>> b = [ 1;  3;  4;6];
>> x=A\b;
[illegible]
>> D=det(A);
```

step 2 [illegible]

```
A =

     [illegible]     1     1
     [illegible]     3     4
     [illegible]     6    10
     [illegible]    10    20

[illegible]
```

[illegible] *A* [illegible] *A* [illegible]

技巧 [illegible] MATLAB [illegible]

3.2.2 奇异线性方程组

例 3.9 求解奇异矩阵的线性方程组解。

Step 1 在 MATLAB 主窗口命令窗口中输入[illegible]

```
>> A = [  1      3      7
-1      4      4
[illegible]
>> b=[ 6;4;15] ;
>> x=A\b
Warning: Matrix is singular to working precision.
x =
   NaN
   Inf
  -Inf
```

说明

[illegible]

Step 2 [illegible]

```
[illegible]
[illegible]
     0
>> rank(A)
ans =
     2
```

提示

[illegible]

[illegible]。其对应的数值解为 pinv(A)*B，下面举例来说明。

例 3.10 使用伪逆矩阵的方法求解奇异矩阵的线性方程的解。

Step 1 [illegible]

```
>> A = [  1      3      7
-1      4      4
1     10     18 ] ;
>> b=[ 5;2;12] ;
>> x=pinv(A)*b;
>> bsol=A*x;
```

Step 2 [illegible]

```
x =
    0.3850
   -0.1103
[illegible]
bsol =
```

```
    5.0000
    2.0000
   12.0000
```

[illegible]

step 3 现在需要求解的数值，在 MATLAB 的命令窗口中输入下面的命令

```
>> A = [illegible]
[illegible]    4    4
1    10    18 ];
>> b=[illegible];
>> x=pinv(A)*b;
>> bsol=A*x;
```

step 4 查看求解的结果，在命令窗口中输入计算的变量名称，例如的结果如下

```
x =
    0.0759
    0.3126
    0.6667
bsol =
    5.6667
    [illegible]
   15.1667
```

[illegible]

3.2.3 欠定线性方程组

在线性代数的定义中，欠定线性方程组是指方程组的未知量个数多于方程个数的问题，这类问题不能求解唯一解，MATLAB 通常给出其中一个基本解，然后用其余零解。为解决欠定问题，MATLAB 采用的是中心解的方法来求解欠定线性方程组。下面举例说明。

例 3.11 求解欠定线性方程组。

step 1 在 MATLAB 的命令窗口中输入下面的命令

```
>> A = [illegible]    2    [illegible]
[illegible]    4    [illegible]    [illegible]
[illegible]    [illegible]    [illegible]
[illegible]    10    11    [illegible] ;
>>b = [ 1;3;5;7] ;
>> [ Q,R] = qr(A) ;
[illegible]
>>xqr = R\y;
>> x=A\b;
```

step 2 查看求解的结果，在命令窗口中输入计算的变量名称，结果如下所示

```
Warning: Rank deficient, rank = 2,  tol =   1.4594e-014.
x =
    0.5000
         [illegible]
    0.1667
Warning: Rank deficient, rank = 2,  tol =   1.4594e-014.
xqr =
    0.5000
         [illegible]
    [illegible]
R =
    [illegible]   [illegible]   [illegible]   [illegible]
   -0.3105   -0.4512   -0.6919    0.4704
   -0.5433   -0.0694   -0.2531   -0.7975
   -0.7762    0.3124    0.3994    0.3748
R =
  -12.8841  -14.5916  -16.2992
         0   -1.0413   -2.0826
         0         0   -0.0000
         0         0         0
```

[illegible]

3.2.4 超定线性方程组

超定线性方程组是指方程组的个数比未知数个数多的情况，对于这种情况，MATLAB提供了两种不同的方法来求解，这两种方法在前面已经介绍过，在本小节中将利用一个具体的实例来说明如何求解超定线性方程组。

例 3.12 求解超定线性方程组。

step 1 在 MATLAB 的命令窗口中输入下面的命令

```
>> A = magic(8);
>>A = A(:,1:6) ;
>> b = 260*ones(8,1);
>> x = pinv(A)*b ;
>>bs=A*x;
>> xs=A\b;
>>bs=A*xs;
>> nx=norm(x);
>> nxs=norm(xs);
```

step 2 查看上面的结果，在命令窗口中输入计算机要显示的，输出的结果如下

```
A =
    64     2     3    61    60     6
     9    55    54    12    13    51
    17    47    46    20    21    43
    40    26    27    37    36    30
    32    34    35    29    28    38
    41    23    22    44    45    19
```

```
    49    15    14    52    53    11
     8    58    59     5     4    62

x =
    1.1538
    1.4615
    1.3846
    1.3846
    1.4615
1.1538

Warning: Rank deficient, rank = 3,  tol =   1.8829e-013.
x =
    4.0000
    5.0000
         ;

         ;
   -1.0000
bsol =
  260.0000
  260.0000
  260.0000
  260.0000
  260.0000
  260.0000
  260.0000
  260.0000
ns =
  260.0000
  260.0000
  260.0000
  260.0000
  260.0000
  260.0000
  260.0000
  260.0000
nx =
    3.2817
nxt =
    [illegible]
```

[illegible]

3.3 矩阵分解

[illegible]

3.3.1 Cholesky 分解

[illegible]，一般应用一个对称正定矩阵 A 的标准[illegible]形式 R 的转置与自身相乘的，即可表示为 $A=R'R$ 的形式[illegible]。其[illegible]，需要进行 Cholesky 分解的矩阵必须是正定的。

在 MATLAB 中，进行 Cholesky 分解的是 chol 命令

- [illegible]（其中 X 为[illegible]），RR [illegible]，[illegible]，使得 $X=R'R$ [illegible] X 为[illegible]定矩阵，该命令会返回错误信息
- [illegible]，[illegible]，其中[illegible]矩阵 R 是一个[illegible]，[illegible]满足等式 $X=R'R$，[illegible]参数 $p=0$ [illegible]，[illegible]参数为正整数，R 为一个[illegible]，[illegible]参数 $p=1$，[illegible]满足 $X(1:p-1,1:p-1)=R'R$。

对对称正定矩阵进行分解在矩阵理论中是个[illegible]，[illegible]分解[illegible]

例 3.13 对对称正定矩阵进行 Cholesky 分解。

step 1 在 MATLAB 命令窗口中输入下面的[illegible]

```
>>X = pascal(n)
>>R = chol(X)
>>[illegible]*R
```

step 2 命令执行完毕，在命令窗口中输入计算的变量名称，回车即可得到[illegible]

```
X =
     1     1     1     1     1
     1     2     3     4     5
     1     3     6    10    15
     1     4    10    20    35
     1     5    15    35    70
R =
     1     1     1     1     1
     0     1     2     3     4
     0     0     1     3     6
     0     0     0     1     4
     0     0     0     0     1

     1     1     1     1     1
           2     3     4     5
           3     6    10    15
     1     4    10    20    35
     1     5    15    35    70
```

[illegible] R [illegible]，[illegible] $R'R=X$ [illegible] Cholesky 分解过程[illegible]

Step 3 修改矩阵结构。在 MATLAB 命令窗口中输入下面命令：

```
>> X(n,n) = X(n,n)-1
>>[ R1,p]=chol(X)
>> C1=transpose(R1)*R1
>> C2=C1(1:p-1,1:p-1)
```

Step 4 查看上面计算结果。在命令窗口中输入计算的变量名称，得到计算结果为：

```
X =
     1     1     1     1     1
     1     2     3     4     5
     1     3     6    10    15
     1     4    10    20    35
     1     5    15    35    69
R1 =
     1     1     1     1
     0     1     2     3
     0     0     1     3
     0     0     0     1
p =
     5
C1 =
     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
C2 =
     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
```

从上面的结果中可以看出，当矩阵不是正定矩阵的情况下，函数 chol 的第二个参数 p 不为 0，矩阵 R 不是正三角阵，并且满足条件 R(1:p-1,1:p-1)=C2

3.3.2 使用 Cholesky 分解求解方程组

例 3.14 使用 Cholesky 分解来求解线性方程组。

Step 1 在 MATLAB 命令窗口中输入下面的命令：

```
>> A=pascal(4);
>>b=[ 1;4;6;13];
>>x=A\b
>>R=chol(A);
>>Rt=transpose(R);
>>xr=R\(Rt\b)R
```

Step 2 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下：

```
x =
```

[illegible]

[illegible]

的结果完全相同。其对应的数学原理如下

[illegible]可以转换为 $R'Rx=b$，该方程组的数值为 $x=R\backslash(R'\backslash b)$。

[illegible]

3.3.3 不完全 Cholesky 分解

[illegible]

- R = cholinc(X,droptol) 其中参数 X 和 R 的含义和 chol 命令中的含义相同，其中 droptol 表示不完全 Cholesky 分解的丢失容限，当该参数为 0 时，则属于完全 Cholesky 分解。
- R = cholinc(X,options) 其中参数 options 用来设置该命令的相关参数。具体地讲，options 是一个结构体，包含 droptol、michol 和 rdiag 三个参数。
- R = cholinc(X,'0') 完全 Cholesky 分解。
- [R,p] = cholinc(X,'0') 和命令 chol(X) 相同。
- R = cholinc(X,'inf') 采用 Cholesky-infinity 方法来进行分解。Cholesky-infinity 方法是基于 Cholesky 分解的，但是可以用来处理实半正定分解。

例 3.15 使用 cholinc 命令对矩阵进行 Cholesky 分解。

Step 1 在 MATLAB 的命令窗口中输入下面的命令。

```
>> H20 = [illegible]
>> [R,p] = chol(H20);
>> Rinf = cholinc(H20,'inf');
>> Rfull=full(Rinf(14:end,14:end))
```

Step 2 查看上面计算结果。命令窗口中输入计算机显示结果，显示结果如下

```
Rfull =
   1.0[illegible]      0      0      [illegible]
   [illegible]   1.0[illegible]      0      0      [illegible]
   [illegible]   [illegible]   1.0[illegible]      0      0      [illegible]
   0      [illegible]      [illegible]   1.0[illegible]      0      0
   [illegible]      [illegible]      [illegible]      [illegible]   1.0[illegible]      0      [illegible]
```

```
[illegible]    0    0    0  Inf    0
[illegible]    [illegible]    [illegible]    [illegible]    0  Inf
```

step 3 检验是否满足分解条件。在 MATLAB 的命令窗口中输入下面的命令

```
>> H=full(H20(14:end,14:end));
>>H20R=Rfull'*Rfull;
```

step 4 查看运算的结果，在命令窗口中输入计算的变量名称，回车的结果如下

```
H =
    0.0370    0.0357    0.0345    0.0333    0.0323    0.0313    0.0303
    0.0357    0.0345    0.0333    0.0323    0.0313    0.0303    0.0294
    0.0345    0.0333    0.0323    0.0313    0.0303    0.0294    0.0286
    0.0333    0.0323    0.0313    0.0303    0.0294    0.0286    0.0278
    0.0323    0.0313    0.0303    0.0294    0.0286    0.0278    0.0270
    0.0313    0.0303    0.0294    0.0286    0.0278    0.0270    0.0263
    0.0303    0.0294    0.0286    0.0278    0.0270    0.0263    0.0256
H20R =
   Inf   NaN   NaN   NaN   NaN   NaN   NaN
   NaN   Inf   NaN   NaN   NaN   NaN   NaN
   NaN   NaN   Inf   NaN   NaN   NaN   NaN
   NaN   NaN   NaN   Inf   NaN   NaN   NaN
   NaN   NaN   NaN   NaN   Inf   NaN   NaN
   NaN   NaN   NaN   NaN   NaN   Inf   NaN
   NaN   NaN   NaN   NaN   NaN   NaN   Inf
```

从上面的结果中，看出，尽管 chol 命令可以求解出对应解结果，但是由于解决问题本身的原因，其结果是不合理的。

3.3.4 LU分解

LU分解又被称为高斯消去法。它可以将任意一个方阵 A 分解为一个"心理"下三角矩阵 L 和一个上三角矩阵 U 的乘积，也就是 $A = LU$。其中，"心理"下三角矩阵的定义为下三角矩阵和置换矩阵的乘积。

在 MATLAB 中，实现 LU 分解的命令为 lu，其主要调用格式如下：

- [L,U] = lu(X)：其中 X 是任意方阵，L 是"心理"下三角矩阵，U 是上三角矩阵，这三个变量满足的条件式为 $X = LU$。
- [L,U,P] = lu(X)：其中 X 是任意方阵，L 是"心理"下三角矩阵，U 是上三角矩阵，P 是置换矩阵，满足的条件式为 $PX = LU$。
- Y = lu(X)：其中 X 是任意方阵，把上三角矩阵和下三角矩阵合并在矩阵 Y 中给出，满足等式为 $Y = L + U - I$，该命令将损失置换矩阵 P 的信息。

例3.16 使用 lu 命令对矩阵进行 LU 分解。

step 1 在 MATLAB 的命令窗口中输入下面的命令

```
>> A = [illegible]  1  [illegible]  [illegible]
 9   -1    2
```

```
          2     -5     7 ];
>> [ L1,U1] =lu(A);
>> A1=L1*U1;
>> x=inv(A);
>> x1=inv(U1)*inv(L1);
>> d=det(A);
>> d1=det(L1)*det(U1);
```

step 2 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下：

```
L1 =
   -0.1111    1.0000         0
    1.0000         0         0
    0.2222   -0.6056    1.0000
U1 =
    9.0000   -1.0000    2.0000
         0    7.8889   -4.7778
         0         0    3.6620
A1 =
    -1     8    -5
     9    -1     2
     2    -5     7
x =
   -0.0115    0.1192   -0.0423
    0.2269   -0.0115    0.1654
    0.1654   -0.0423    0.2731
x1 =
   -0.0115    0.1192   -0.0423
    0.2269   -0.0115    0.1654
    0.1654   -0.0423    0.2731
d =
  -260
d1 =
  -260
```

从上面的结果可以看出，方阵的LU分解满足下面的等式条件：

$$A=LU、U^{-1}L^{-1}=A^{-1}\text{ 和 }\det(A)=\det(L)\det(U)$$

step 3 使用三个输出变量的命令形式。在MATLAB的命令窗口中输入下面的命令：

```
>> [ L,U,P] = lu(A);
>> Lp=P*L;
>> Ap=L*U;
>> Pa=P*A;
```

step 3 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下：

```
L =
    1.0000         0         0
   -0.1111    1.0000         0
    0.2222   -0.6056    1.0000
U =
    9.0000   -1.0000    2.0000
         0    7.8889   -4.7778
         0         0    3.6620
```

```
P =
     0     1     0
     1     0     0
     [illegible]
Ap =
     [illegible]    -1     2
    -1     8    -5
     [illegible]
PA =
     9    -1     2
    -1     8    -5
     2    -5     7
Lp =
   -0.1111    1.0000         0
    1.0000         0         0
    0.2222   -0.6056    1.0000
```

从上面的结果可以看出，使用一个输出变量的命令满足下面的等式关系

$$PA=LU \text{ 和 } PL=L'$$

在上面的等式中，L'表示使用两个输出变量求解的LU分解矩阵结果。

step 5 使用LU分解来求解线性方程组。在MATLAB的命令窗口中输入下面的命令

```
>>b=[1;2;5];
>>xb=A\b;
>>y1 = L\b;
>>xb1=U\y1;
>> y2 = L1\b;
>>xb2=U1\y1;
```

step 6 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下

```
xb =
    0.[illegible]
    1.[illegible]
    1.[illegible]
xb1 =
    0.[illegible]
    1.[illegible]
    1.[illegible]
xb2 =
    0.[illegible]
    1.[illegible]
    1.[illegible]
```

说明

从上面的结果可以看出，使用两个输出变量还是一个输出变量的命令求解的结果，都会得出在命令中得到的结果相同。

5.3.5 不完全LU分解

对于稀疏矩阵，MATLAB提供函数luinc来进行不完全的LU分解。其调用格式如下

- [illegible] = lunc(X,opt,pcl) 命令中参数[illegible]lu[illegible]，其中[illegible]表示[illegible]个全LU分解的主[illegible]表，[illegible]参数[illegible]，[illegible]解
- [illegible] = lunc(X,opt,[illegible]) 参数opt的设置[illegible]，[illegible]参数
- [L,U] = luinc(X,'0') 0级不完全LU分解
- [L,U,P] = luinc(X,'0') 0级不完全LU分解。

例3.17 使用lunic命令对稀疏矩阵进行LU分解。

step 1 加载稀疏矩阵，并绘制稀疏矩阵图形。在MATLAB命令窗口，输入下面的语句：

```
% 加载稀疏矩阵
>> load west0479;
>> S = west0479;
>> LU = lu(S);
% 绘制稀疏矩阵的图形
>> subplot(1,2,1);
>> spy(S);
[illegible]
% 使用LU求解得到的结果
>> subplot(1,2,2);
>> spy(LU);
[illegible]
```

step 2 查看上面的结果，在命令窗口输入[illegible]，按"Enter"键，则可以看到如图3.1所示。

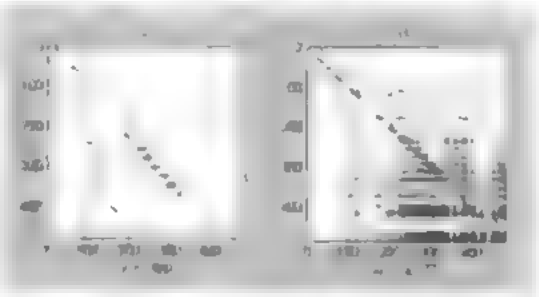


图3.1 原数矩阵和LU分解结果图形

由于加载的原数矩阵维数比较大，如果直接使用命令行语句，[illegible]。在上面的实例中，使用了spy命令来查看矩阵的稀疏性。

step 3 使用lunic命令对稀疏矩阵进行LU分解。在命令窗口输入下面的语句：

```
% 进行0级LU分解
>> [L,U,P] = luinc(S,'0');
>> D = (L*U).*spones(P*S)-P*S;
% 打开新的绘图窗口
% 绘制使用lunic命令得到的结果
[illegible]
>>subplot(221);
>>spy(L);
[illegible]
>> subplot(222);
>>spy(U);
```

```
[illegible]
[illegible]
>>spy(P*S);
[illegible]
[illegible]
[illegible]
[illegible]
```

step 4 查看本例结果，在输入上面文件和结果后，按"Enter"键，得到的图形如图3.2。

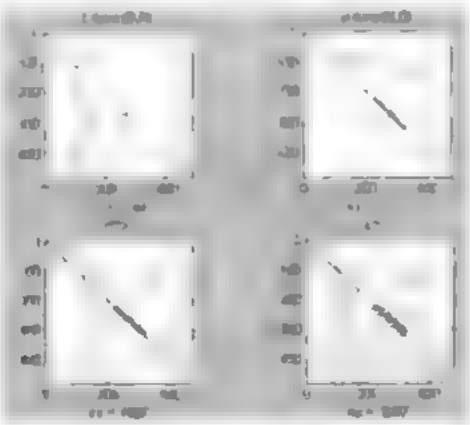


图3.2 使用lunic命令得到的结果

step 5 使用不完全带阈值的稀疏矩阵阵进行求解，在命令窗口输入上面命令：

```
>>[ IL1,IU1,IP1] = luinc(S,1e-8);
>> [ IL2,IU2,IP2] = luinc(S,1e-4);
Warning: Incomplete upper triangular factor has zero diagonal.
         It cannot be used as a preconditioner for an iterative method
>> [ IL3,IU3,IP3] = luinc(S,1e-2);
Warning: Incomplete upper triangular factor has zero diagonal.
         It cannot be used as a preconditioner for an iterative method
>> [ IL4,IU4,IP4] = luinc(S,1);
Warning: Incomplete upper triangular factor has zero diagonal.
         It cannot be used as a preconditioner for an iterative method
>> figure;
>> subplot(221);
>> spy(IL1*IU1);
>> title('luinc(S,1e-8)')
>> subplot(222);
[illegible]
title('luinc(S,1e-4)')
[illegible]
spy(IL3*IU3);
[illegible]
>> subplot(224);
[illegible]
[illegible]
```

step 6 查看求解结果，在输入上面文件和结果后，按"Enter"键，得到的图形如图3.3。

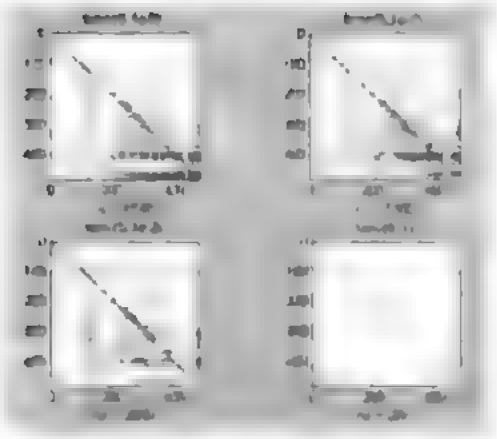


图3.3 使用不同的误差容忍度

step 1 绘制 'droptolerance' 图中，在本实例中，droptol表示一个稀疏矩阵的丢失容限，nnz表示系数矩阵中非零元素的个数，由此可见，这两个变量之间的关系，在MATLAB的命令窗口中输入下面的代码。

```
>> nz(1)=nnz(LL1);
[illegible]
[illegible]
tol(1)=1.e-8;
[illegible]
[illegible]
[illegible]=nnz([illegible]);
[illegible]
[illegible]
tol(4)=1;
>> semilogx(tol,nz,'g','LineWidth',1.5)
[illegible]
set(gca,'Ylim',[0 7800]);
>> title('Drop tolerance vs nnz(luinc(S,droptol))')
[illegible]
[illegible]
>> grid
```

step 2 查看本例的结果，右键单击[illegible]，选择'[illegible]'项，在弹出的菜单中选择'[illegible]'。

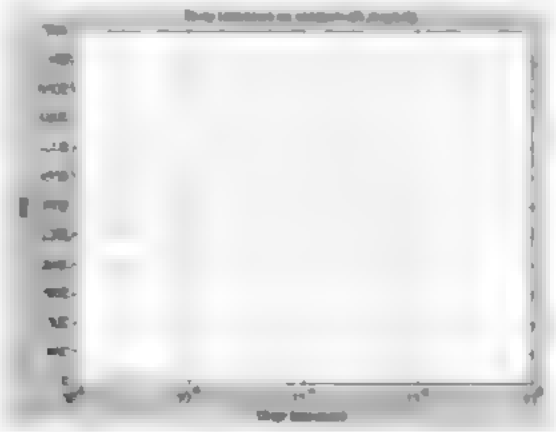


图3.4 丢失容限和nnz的关系

Step 9 [illegible] 在命令中，droptol 表示 ILU 分解的丢失容限，[illegible] 的关系，[illegible] 的代码

```
%计算相对误差
>> normlu(1)=norm(IL1*IU2-IP1*S,1)/norm(S,1);
>> normlu(2)=norm(IL2*IU2-IP2*S,1)/norm(S,1);
>> normlu(3)=norm(IL3*IU3-IP3*S,1)/norm(S,1);
>> title('Drop tolerance vs norm norm(L*U-P*S,1)/norm(S,1)')
>>xlabel('Drop tolerance')
>> grid
```

Step 10 在命令解决环境中，[illegible]，按 "Enter" 键，[illegible]

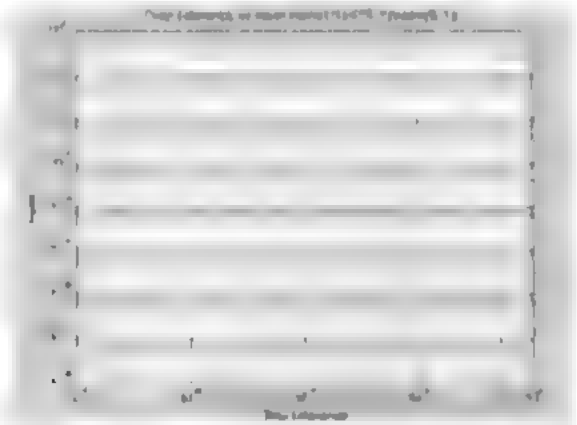


图 3.5 丢失容限和相对误差的图形

3.3.6 QR 分解

[illegible] 解，[illegible] $m \times n$ 的矩阵 A 分解为一个正交矩阵 Q 和一个上三角矩阵 R 的乘积，也就是说 $A=QR$。

在 MATLAB 中，进行 QR 分解的命令为 qr。其调用格式如下

- [illegible] R 和矩阵 A 的大小相同，Q 是一个正交矩阵，满足等式 $A=QR$。该调用方式适用于满矩阵和稀疏矩阵
- [illegible] 分解，[illegible] A 是一个 $m \times n$ 的矩阵，其中 $m>n$，[illegible] 为 m 行 n 列的矩阵，R 是一个方阵 R 是 $n \times n$ 的矩阵，如果 $m \le n$，[illegible] 命令[Q,R] = qr(A)相同。该调用方式适用于满矩阵和稀疏矩阵
- [illegible]，Q 是一个正交矩阵，R 是一个上三角矩阵，E 是一个置换矩阵，满足等式 $A \cdot E=Q \cdot R$。该调用方式适用于满矩阵。

例 3.18 使用 qr 命令对矩阵进行 QR 分解。

Step 1 在 MATLAB 命令窗口中，输入下面的命令

```
>> A = [1 2 3;4 5 6;7 8 9]
```

```
[ Q,R] = qr(A)
C=Q*R
```

step 2 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下：

```
A =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
Q =
   -0.5234    0.5058    0.6735   -0.1215   -0.0441
   -0.7081   -0.6966   -0.0177    0.0815   -0.0800
   -0.1231    0.1367   -0.3558   -0.6307   -0.6646
   -0.3079    0.1911   -0.4122   -0.4247    0.7200
   -0.3387    0.4514   -0.4996    0.6328   -0.1774
R =
  -32.4808  -26.6311  -21.3973  -23.7063  -25.8615
         0   19.8943   12.3234    1.9439    4.0856
         0         0  -24.3985  -11.6316   -3.7415
         0         0         0  -20.0982   -9.9739
         0         0         0         0  -16.0005
C =
   17.0000   24.0000    1.0000    8.0000   15.0000
   23.0000    5.0000    7.0000   14.0000   16.0000
    4.0000    6.0000   13.0000   20.0000   22.0000
   10.0000   12.0000   19.0000   21.0000    3.0000
   11.0000   18.0000   25.0000    2.0000    9.0000
```

从上面的结果可以看出，矩阵R是上三角矩阵，同时满足等式$A=QR$，在下面的步骤中，将需要证明Q矩阵是正交矩阵。

step 3 证明矩阵Q的正交性。在 MATLAB 的命令窗口中输入下面的命令：

```
>> detQ=det(Q);
>> for i=1:4
A=Q(:,i);
for j=(i+1):5
B=Q(:,j);
C=A'*B;
disp(num2str(C))
end
end
```

step 4 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下：

```
detQ =
    1.0000
C=
5.5511e-017
5.5511e-017
-2.7756e-017
6.9389e-018
```

```
0
[illegible]

-6.9399e-017
[illegible]
```

[illegible] Q [illegible] 0，因此 Q 是正交矩阵。

[illegible] Q 是正交矩阵，可以证明 Q 的逆矩阵 Q^{-1} 和原矩阵 A 是等价的，因此可以通过分析 R 的性质来判断 A 的性质。

3.3.7 操作 QR 分解结果

在 MATLAB 中，除了提供 qr 命令外，还提供 qrdelete 和 qrinsert 命令来处理矩阵 A 的 QR 分解。其中，qrdelete 可以删除矩阵 A 的第 j 列，qrinsert 则可以插入 A 的第 j 列。[illegible]

- [illegible] 矩阵 A 的 QR 分解结果，其中 A_1 为删除矩阵 A 的第 j 列得到的结果，而矩阵 $A = QR$。
- [illegible] 矩阵 A 的 QR 分解结果，其中 A_1 为在矩阵 A 的第 j 列插入 [illegible] 得到的结果，而矩阵 $A = QR$。
- [illegible] 矩阵 A 的 QR 分解结果，其中 A_1 为删除矩阵 A 的第 j 行的数据得到的结果，而矩阵 $A = QR$。

例 3.19 对矩阵 QR 分解得到的矩阵进行删除运算。

Step 1 [illegible]

```
A = magic(5);
[Q,R] = qr(A);j = 3;
[Q1,R1] = qrdelete(Q,R,j,'row');
C=Q1*R1;
[illegible]
[illegible]
```

Step 2 [illegible]

```
[illegible]
    0.7135    0.6911    0.0158    0.1147
    0.3102   -0.1982    0.4675   -0.8037
    0.3413   -0.4616    0.5768    0.5811

   12.2355   26.0918   19.9482   21.4063   23.3297
         0  -19.7045  -10.9891    0.4318   -1.4873
         0         0   22.7444    5.8357   -3.1977
         0         0         0  -14.5784    3.7796

   17.0000   24.0000    1.0000    8.0000   15.0000
```

```
   23.0000     5.0000     7.0000    14.0000    16.0000
   10.0000    12.0000    19.0000    21.0000     3.0000
   11.0000    18.0000    25.0000     2.0000     9.0000
A2 =
    17    24     1     8    15
    23     5     7    14    16
    10    12    19    21     3
    11    18    25     2     9
```

[illegible] C Q R [illegible] A [illegible]
面结果中的 A_1 矩阵。

step 3 [illegible] Q [illegible]

```
>> detQ1=det(Q1);
[illegible]
[illegible]
for j=(i+1):4
B=Q1(:,j);
C=A'*B;
[illegible]
[illegible]
end
```

step 4 [illegible]

```
detQ1 =
    1.0000
[illegible]
1.1102e-016
[illegible]
```

[illegible] Q [illegible]
为 0，因此 Q_1 是正交矩阵。

[illegible]

例 3.20　对矩阵 QR 分解得到的矩阵进行插入运算。

step 1 [illegible]

```
>> A = magic(5);
[Q,R] = qr(A);
[illegible]
[illegible]
[Q1,R1] = qrinsert(Q,R,j,x,'row');
[illegible]
>> A2 = [A(1:j-1,:); x; A(j:end,:)];
```

step 2 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下：

```
Q1 =
    0.5231    0.5039   -0.6750    0.1205    0.0411    0.0225
    0.7078   -0.6966    0.0190   -0.0788    0.0833   -0.0150
    0.0308    0.0592    0.0656    0.1169    0.1527   -0.9769
    0.1231    0.1363    0.3542    0.6222    0.6398    0.2104
    0.3077    0.1902    0.4100    0.4161   -0.7264   -0.0150
    0.3385    0.4500    0.4961   -0.6366    0.1761    0.0225
R1 =
   32.4962   26.6801   21.4795   23.8182   26.0031
         0   19.9292   12.4403    2.1340    4.3271
         0         0   24.4514   11.8132    3.9931
         0         0         0   20.2382   10.3392
         0         0         0         0   16.1948
         0         0         0         0         0
Aqr =
   17.0000   24.0000    1.0000    8.0000   15.0000
   23.0000    5.0000    7.0000   14.0000   16.0000
    1.0000    2.0000    3.0000    4.0000    5.0000
    4.0000    6.0000   13.0000   20.0000   22.0000
   10.0000   12.0000   19.0000   21.0000    3.0000
   11.0000   18.0000   25.0000    2.0000    9.0000
A2 =
    17    24     1     8    15
    23     5     7    14    16
     1     2     3     4     5
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

从上面的结果中可以看出，在上面的结果中，满足等式$Aqr=Q_1 \cdot R_1$，其中Aqr就是矩阵A删除对应数据行的结果，也就是上面结果中的A_2矩阵。

step 3 证明Q_1矩阵的正交性。在MATLAB的命令窗口中输入下面的命令：

```
>> detQ1=det(Q1);
for i=1:5
A=Q1(:,i);
for j=(i+1):6
B=Q1(:,j);
C=A'*B;
disp(num2str(C))
end
end
```

step 4 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下：

```
detQ1 =
    1.0000
C=
-5.5511e-017
5.5511e-017
0
-6.9389e-018
```

```
-4.3368e-018
-5.5511e-017
0
-1.3878e-017
-1.7347e-018
0
0
 3.9899e-017
-2.6368e-016
-4.8572e-017
-3.9031e-017
```

奇异值分解

奇异值分解在矩阵分析中有着重要的地位，对于任意矩阵 $A \in C^{m\times n}$，存在酉矩阵（Unitray matrix），$U=[u_1, u_2, \cdots, u_n]$，$V=[v_1, v_2, \cdots, V_n]$。使得

$$U^T AV = \mathrm{diag}(\sigma_1, \sigma_2, \cdots, \sigma_p)$$

其中参数 $\sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_p, p=\min\{m,n\}$。在上面的式子中，$\{\sigma_i, u_i, v_i\}$分别是矩阵$A$的第$i$个奇异值、左奇异值和右奇异值，它们的组合就称为奇异值分解三对组。

在 MATLAB 中，计算奇异值分解的命令如下：

- [U,S,V] = svd(X)　奇异值分解；
- [U,S,V] = svd(X,0)　比较经济的奇异值分解；
- s = svds(A,k,0)　向量 s 中包含矩阵 A 分解得到的 k 个最小奇异值；
- [U,S,V] = svds(A,k,0)　给出 A 的 k 个最大奇异值分解结果。

例 3.21　对矩阵进行奇异值分解。

step 1 在 MATLAB 的命令窗口中输入下面的命令：

```
>> X=[ 1     2
       3     4
       5     6
       7     8] ;
>> [ U,S,V]  = svd(X)
```

step 2 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下：

```
U =
   -0.1525   -0.8226   -0.3945   -0.3800
   -0.3499   -0.4214    0.2428    0.8007
   -0.5474    0.0201    0.6979   -0.4614
   -0.7448    0.3812   -0.5462    0.0407
S =
   14.2691         0
         0    0.6268
         0         0
         0         0
V =
   -0.6414    0.7672
   -0.7672   -0.6414
```

Step 3 [illegible]

```
>> X =
     3     4
     5     6
     7     8];
>> [U,S,V] = svd(X,0)
```

Step 4 [illegible]

```
    -0.1525   -0.9226
    -0.3499   -0.4214
    -0.5474   -0.0201
    -0.7448    0.3812

    14.2691         0
          0    0.6268
V =
    -0.6414    0.7672
    -0.7672   -0.6414
```

例 3.22 使用 svd 和 svds 命令对稀疏矩阵进行奇异值分解。

Step 1 在 MATLAB 的命令窗口中输入下面的命令

```
>> load west0479
>> [illegible](west0479);
s1 = svds(west0479,4);s8 = svds(west0479,6,0);
```

Step 2 查看计算结果，在命令窗口中输入计算变量名称，得到如下结果。

```
>> [illegible]
  1.0e+005 *
    3.1895
    3.1725
    3.1695
    3.1685
>> [illegible]
  1.0e-004 *
    0.5616
    0.5169
    0.4505
    0.4020
    0.0424
    0.0098
```

[illegible]

Step 3 绘制数值结果函形。在 MATLAB 的命令窗口中输入下面的命令

```
>> plot(s1,'ro')
```

```
>> hold on
>> plot(s,'gp')
>> set(gca,'XtickL',[0:1:5])
>> set(gca,'Xlim',[0 4.5])
>> xlabel('n')
```

step 6 [illegible]

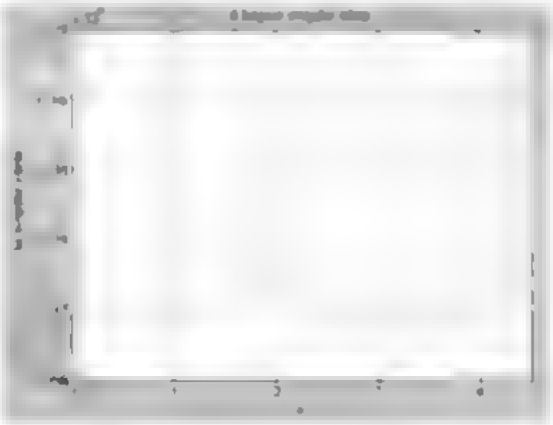


图3.6 最大4个奇异值

说明 从上面的结果来看，[illegible]

3.4 特征值分析

在线性代数的理论中，对于 $n \times n$ 的 A，其特征值 λ 和特征向量 x 满足下面的等式

$$Ax=\lambda x$$

在上面的等式中，λ 是一个标量，x 是一个向量。矩阵 A 有 n 个特征值，[illegible] 对角线上的元素是一个矩阵 D，[illegible] $D=\mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$。[illegible] 特征值对应的特征向量，[illegible] 矩阵 V 的数据列。如果该矩阵 V 是可逆的，[illegible] 特征值问题可以描述为

$$A \cdot V=V \cdot D \Rightarrow A=V \cdot D \cdot V^{-1}$$

在 MATLAB 中，要计算矩阵的特征值和特征向量，[illegible] 特征值的多种内容，下面分小节详细分析。

3.4.1 特征值和特征向量

在 MATLAB 中，[illegible] 特征值和特征向量的函数 [illegible] 矩阵 [illegible] 特征值和特征向量，[illegible] 下面一起来看看。

- d = eig(A)：仅计算矩阵 A 的特征值，并以向量的形式输出。
- [V,D] = eig(A)：计算矩阵 A 的特征向量矩阵 V 和特征值对角阵 D，满足等式 $AV=VD$。

- [illegible] A,'nobalance') [illegible] 矩阵 A 中有数据[illegible]，[illegible]
- [illegible] = eig(A,B) [illegible] 计算矩阵 A 的[illegible]特征值对角阵 D，满足等式 $AV=BVD$
- [illegible] = eig(A,B,sigma) 计算[illegible]矩阵 A 和 B 的[illegible]特征值。关于参数 sigma 的取值，请查看相应的帮助文件。

[illegible]

例 3.23 对基础处理求解矩阵的特征值和特征向量。

step 1 对[illegible]特征值分析。在 MATLAB 的命令窗口中输入，[illegible]

```
>> A=pascal(5);
>> [illegible](A)
```

step 2 查看[illegible]结果。在命令窗口中输入计算的变量名称，得到的结果如

```
V =
    0.1680   -0.5706   -0.7660    0.2429    0.0175
   -0.5517    0.5587   -0.3830    0.4808    0.0749
    0.7025    0.2529    0.1642    0.6110    0.2055
   -0.4071   -0.5179    0.4377    0.4130    0.4515
    0.0900    0.1734   -0.2189   -0.4074    0.8649
D =
    0.0108         0         0         0         0
         0    0.1812         0         0         0
         0         0    1.0000         0         0
         0         0         0    5.5175         0
         0         0         0         0   92.2904
```

step 3 检验分析得到的结果。在 MATLAB 的命令窗口中输入下面的命令

```
>> detV=det(V);
>> S=A*V-V*D;
```

step 4 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下

```
detV =
    1.0000
S =
  1.0e-015 *
    0.0620   -0.0154    0.0304    0.1110         0
    0.1057   -0.0271    0.0694    0.0833         0
    0.0649   -0.0717    0.0900   -0.0555    0.1110
    0.0684   -0.0283    0.0416    0.1527    0.1110
    0.0753    0.0933   -0.0191    0.0555    0.1110
```

[illegible] $AV=VD$

例 3.24 [illegible]

STEP 1 对矩阵进行特征值分析，在 MATLAB 的命令窗口中输入下面的命令：

```
>> B = [ 3      -4      -1.3      [illegible]
    -2       3       [illegible]      -eps/5
    -eps/3   eps/2   -1       [illegible]
    -1.5     -.5      .3      1      [illegible]
>> [VB,DBJ] = eig(B);
>> ER1=B*VB - VB*DB;
>> [VN,DNJ] = eig(B,'nobalance');
>> ER2=B*VN - VN*DN;
```

[illegible]

STEP 2 查看本例的结果，在命令窗口中输入计算机显示的结果，得到如下结果：

```
VB =
      [illegible]    -0.3926    [illegible]    0.0420
   -0.5668    -0.2772    0.0000    0.4408
   -0.0000    -0.0000    0.0000   -0.8396
   -0.1903    -0.8772    1.0000   -0.3148
VN =
    1.0000    -0.4469   -0.0000    0.0500
   -0.7071    -0.3160   -0.0000    0.5250
   -0.0000    -0.0000   -0.0000   -1.0000
   -0.2374    [illegible]    [illegible]    [illegible]

DB =
    5.8284         0         [illegible]
         0    [illegible]         [illegible]
         0         [illegible]    [illegible]
         0         [illegible]    [illegible]    [illegible]
DN =
    [illegible]         0         [illegible]
         0    [illegible]         [illegible]
         0         [illegible]    [illegible]
         0         [illegible]    [illegible]    [illegible]

ER1 =
    [illegible]    [illegible]    0.0000    [illegible]
    [illegible]    [illegible]   -0.0000    [illegible]
    [illegible]    [illegible]    0.0000    [illegible]
    [illegible]    [illegible]   -0.0000    [illegible]
ER2 =
  1.0e-014 *
    0.1776    0.0514    0.0290    [illegible]
    0.0888   [illegible]   -0.0173    [illegible]
    0.0006    [illegible]    0.0020    [illegible]
         0    [illegible]         0         0
```

从上面的结果可以看出，如果矩阵中的元素和数据元素相当多，如果在命令中采用"nobalance"参数，则会影响结果的精确度。对矩阵A在自程序中的中间计算结果为且结果中包含[illegible]的情况，就会发生上述的情况。

3.4.2 稀疏矩阵的特征值和特征向量

例3.25 使用eigs函数求解稀疏矩阵的特征值和特征向量。

step 1 生成稀疏矩阵，并求出特征值。在MATLAB的命令窗口中输入下面的命令。

```
>>A = delsq(numgrid('C',30));
>>d = eig(full(A));
>>[dum,ind] = sort(abs(d));
>>dlm = eigs(A);
>>dsm = eigs(A,6,'sm');
>>dsmt=sort(dsm);
>>subplot(2,1,1)
>>plot(dlm,'r+')
>>hold on
>>plot(d(ind(end:-1:end-5)),'ro')
>>hold off
>>legend('eigs(A)','eig(full(A))',3)
>>set(gca,'XLim',[0.5 6.5])
>>grid
>>title('Six largest magnitude eigenvalues')
>>subplot(2,1,2)
>>plot(dsmt,'r+')
>>hold on
>>plot(d(ind(1:6)),'ro')
>>hold off
>>legend('eigs(A,6,''sm'')','eig(full(A))',2)
>>grid
>>set(gca,'XLim',[0.5 6.5])
>>title('Six samllest magnitude eigenvalues')
```

step 2 查看求解的结果。在输入上面的程序代码后，按"Enter"键，得到的图形如图3.7所示。

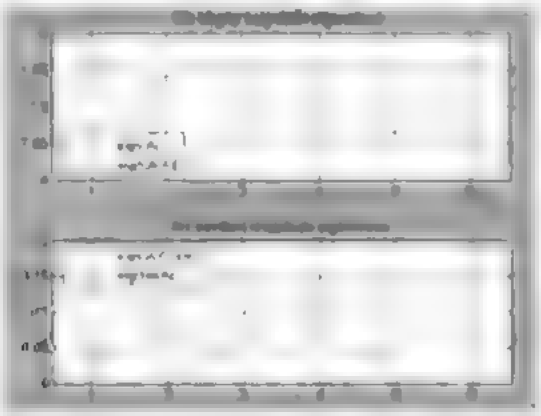


图3.7 计算的图形结果

[illegible]

3.4.3 特征值问题的条件数

[illegible]

$$\frac{1}{\cos(\theta)}$$

[illegible]

在 MATLAB 中，计算特征值条件数的命令如下

- [illegible]
- [illegible]

例 3.26 [illegible]

step 1 在 MATLAB 的命令窗口中输入下面的命令

```
>>A=magic(10);
[illegible]
>>ceig=condeig(A);
```

step 2 查看计算结果，在命令窗口中输入计算的结果变量名称，按下回车键。

```
cequ =
  2.0660e+018
ceig =
    1.0000
    1.0261
    1.9128
    2.7390
    [illegible]0802
    [illegible].8862
    [illegible]8891
    [illegible]3247
    [illegible]4247
    [illegible].6460
```

[illegible]

step 3 要知道矩阵复杂，进入分析，在 MATLAB 的命令窗口中输入下面的命令

```
>> A=eye(5,5);
>> A(3,2)=1;
>>A(2,5)=1;
>> cequ=cond(A);
```

```
>> [c,q]=condeig(A);
```

step 4 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下

```
Warning: Matrix is close to singular or badly scaled.
         Results may be inaccurate. RCOND = 2.465190e-032.
A
     1     0     0     0     0
     0     1     0     0     1
     0     1     1     0     0
     0     0     0     1     0
     0     0     0     0     1
cequ =
    1.[illegible]
ceig =
  1.0e+031 *
    0.0000
    0.0000
    2.0[illegible]
    0.0000
    2.0282
```

[illegible]

例3.27 对与换矩阵进行条件数分析。

step 1 在MATLAB的命令窗口中输入下面的命令

```
>> A=gallery(5);
>> [V,D,c_eig]=condeig(A)
>> condA=cond(A)
```

step 2 查看求解的结果。在命令窗口中输入计算的变量名称，得到的结果如下

```
V =
 0.0000        -0.0000 + 0.0000i  -0.0000 - 0.0000i
                0.0000 + 0.0000i   0.0000 - 0.0000i
 0.0206         0.0206 + 0.0001i   0.0206 - 0.0001i
                0.0207 + 0.0001i   0.0207 - 0.0001i
 0.1398        -0.1397 + 0.0001i  -0.1397 - 0.0001i
               -0.1397 + 0.0000i  -0.1397 - 0.0000i
-0.9574         0.9574             0.9574
                0.9574             0.9574
-0.2519         0.2519 - 0.0000i   0.2519 + 0.0000i
                0.2519 - 0.0000i   0.2519 + 0.0000i

D =
  -0.0408            0                  0                  0                  0
        0      -0.0119 + 0.0386i        0                  0                  0
        0            0            -0.0119 - 0.0386i        0                  0
        0            0                  0        0.0323 + 0.0230i             0
        0            0                  0                  0        0.0323 - 0.0230i
c_eig =
```

```
  1.0e+010 *
    2.1293
    [illegible]
    2.0796
    2.0020
    [illegible]
condA =
  2.0253e+018
```

在上面的例子中，矩阵A的条件数很大，说明它接近奇异。用它来求解线性方程组，可能是一个不稳定的问题，对于数据的微小扰动，或矩阵元素的舍入误差，会使解的结果发生很大变化。

3.4.4 特征值的复数问题

在实际应用中经常遇到复数矩阵，其对应的特征值和特征向量都是复数。在实际应用中，经常需要将一对共轭复数特征值转换为一个实数块，为此MATLAB提供了下面的命令。

- [illegible] = cdf2rdf(VC,DC)：把复数对角矩阵转换为实数对角阵。
- [illegible] = cdf2rdf(VC,DR)：把实数对角阵转换成复数对角阵。

在上面的命令中，VC表示全复数对特征值对角阵，DC表示对应的特征向量矩阵；DR表示实数块特征值对角阵，VR表示对应的实特征向量矩阵。

例3.28 对矩阵的复数特征值进行分析。

step 1 在MATLAB的命令窗口中输入下面的命令。

```
  [illegible]  2   3
  [illegible]  [illegible]
  [illegible]  [illegible]
>> [VC,DC] = eig(X);
>> [VR,DR] = cdf2rdf(VC,DC);
>> XR=VR*DR/VR;
>> X-XR*[illegible];
```

step 2 查看不同的结果，在命令窗口中输入以下语句查看，将会显示如下。

```
VC =
   1.0000              -0.0191 - 0.4002i  -0.0191 + 0.4002i
        0                   0 - 0.6479i        0 + 0.6479i
                         0.6479             0.6479
DC =
   1.0000                   0                  0
        0              4.0000 + 5.0000i         0
                          [illegible]        [illegible]
VR =
    1.0000   -0.0191   -0.4002
         0         0   -0.6479
         0    0.6479         0
DR =
```

```
    1.0000         0         0
         0    4.0000    5.0000
         0   -5.0000    4.0000
XC =
   1.0000              2.0000 + 0.0000i   3.0000
        0              4.0000             5.0000
        0             -5.0000             4.0000
XR =
    1.0000    2.0000    3.0000
         0    4.0000    5.0000
         0   -5.0000    4.0000
```

函数的零点

对于某任意函数，在求解范围之内可能有零点，也可能没有零点；可能只有一个零点，也可能有多个甚至无数个零点。因此，这就给程序求解函数的零点增加了很大的难度，没有可以求解所有函数零点的通用求解命令。本节将简单讨论一元函数和多元函数的零点求解问题。

一元函数的零点

在所有函数中，一元函数是最简单的，同时也是可以使用MATLAB提供的图形绘制命令来实现可视化的。因此，在本小节中将首先讨论一元函数零点的求取方法。

在MATLAB中，求解一元函数零点的命令是fzero，其调用格式如下：

- x = fzero(fun,x0)　参数fun表示的是一元函数，x0表示求解的初始数值；
- [x,fval,exitflag,output] = fzero(fun,x0,options)：参数options的含义是指优化迭代所采用的参数选项，该参数和后面章节中需要讲解到的fsolve、fminbnd、fminsearch等命令中的options都是相同的"模块"；在输出参数中，fval表示对应的函数值，exitflag表示程序退出的类型，output则反映优化信息的变量。

例3.29　求函数$f(x)=x^2\sin x-x+1$在数值区间[-3,4]中的零点。

step 1 绘制函数的图形。在MATLAB的命令窗口中输入下面的命令：

```
%计算函数数值
>> x=[-3:0.1:4];
>>y=sin(x).*x.^2-x+1;
%绘制函数图形
>>plot(x,y,'r','LineWidth',1.5)
>>hold on
%添加水平线
>>h=line([-3,4],[0,0]);
%设置直线的宽度和颜色
>>set(h,'LineWidth',1.5)
>>set(h,'color','k')
%设置坐标轴刻度
>>set(gca,'Xtick',[-3:0.5:4])
%添加图形标题和坐标轴名称
>>title('The zero of function')
>> grid
>> xlabel('x')
```

```
>> ylabel('f(x)')
```

step 2 [illegible]

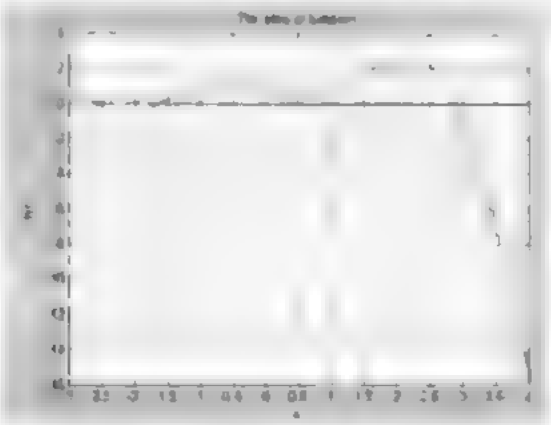


图3.8 函数的图形

提示 [illegible] fzero 命令时，要给定搜索的初始值 x0。

step 3 [illegible]

```
>> [x1,f1,exitflag1]=fzero(f,-2.5);
>>[x2,f2,exitflag2]=fzero(f,-1.5);
[illegible]
[illegible] x=[x1,x2,x3]
>> f=[f1,f2,f3]
```

step 4 [illegible]

```
x =
   -2.5708   -1.6194    2.9142
f =
  1.0e-015 *
   -0.8882    0.2220   -0.8882
```

[illegible] 函数 $f(x)=x\sin x-x+1$ 在[-3,3][illegible]为 -2.5708、-1.6194 和 2.9142。

提示 对于一[illegible]函数，MATLAB 提供 roots 命令求解多项式函数的所有零点，[illegible]的基本原理是求解多项式对应的伴随矩阵的特征值来解的。

3.5.2 多元函数的零点

[illegible]

在 MATLAB 中，求解多元函数方程、组 fsolve，其语法格式如下。

- x = fsolve(fun,x0)　解非线性方程组的数值解
- [x,fval,exitflag,output] = fsolve(fun,x0,options)　指定选项，返回函数在 x 处求值的函数。

例 3.30　求解方程组 $\begin{cases} 2x_1 - x_2 = e^{-x_1} \\ -x_1 + 2x_2 = e^{-x_2} \end{cases}$ 的零点。

step 1　为方程组的求解，在 MATLAB 的命令窗口中输入下面的命令

```
%创建二维图形的数据网格
x=[-10:0.1:10];
y=x;
[X,Y]=meshgrid(x,y);
%计算一维函数的数值
Z=2*X-Y-exp(-1*X);
%绘制曲面图
surf(X,Y,Z)
%设置图形属性
shading interp
%添加水平的颜色条
colorbar horiz
%设置图形的坐标轴和显示属性
set(gca,'Ztick',[-180:20:20])
set(gca,'ZLim',[-170 20])
%设置透明属性
alphamap('rampdown')
colormap hot
%添加图形标题和坐标轴名称
title('The figure of the function')
xlabel('x')
ylabel('y')
zlabel('z')
```

step 2　按照步骤一，在命令窗口中输入下面的命令，按下 Enter 键，即可绘制函数图形，如图 3.9 所示。

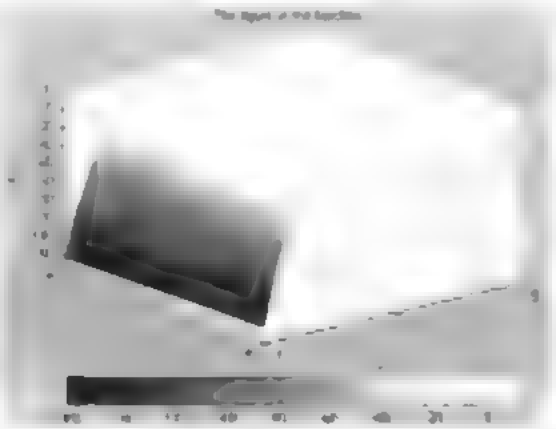


图 3.9　函数图形

step 3　编辑主函数的 M 文件。选择菜单栏中的"File"→"New"→"M-file"命令，打开 M 文件编辑器，在其中输入下面的程序代码

```
function F = fsolvefun (x)
F = [ 2*x(1) - x(2) - exp(-x(1));
      -x(1) + 2*x(2) - exp(-x(2))];
```

在输入上面的程序代码后，将代码保存为 'fsolvefun.m' 文件。

step 4 下面求解方程组，在MATLAB的命令窗口中输入下面的代码：

```
x0 = [ -5; -5];
options=optimset('Display','iter');
[ x,fval] = fsolve(@fsolvefun,x0,options)
```

step 5 查看求解结果，在输入上面的代码后，按下"Enter"键，得到如下结果：

```
                                        Norm of       First-order   Trust-region
Iteration  Func-count     f(x)          step          optimality    radius
     0          3          47071.2                      2.29e+004
     1          6          12003.4             1        5.75e+003          1
     2          9          3147.02             1        1.47e+003          1
     3         12          854.452             1             388           1
     4         15          239.527             1             107           1
     5         18          67.0412             1            30.8           1
     6         21          16.7042             1            9.05           1
     7         24          2.42788             1            2.26           1
     8         27         0.032658      0.759511           0.206         2.5
     9         30      7.03149e-006     0.111927         0.00294         2.5
    10         33      3.29525e-013   0.00169132      6.36e-007         2.5
Optimization terminated: first-order optimality is less than options.
TolFun.
x =
    0.5671
    0.5671
fval =
  1.0e-006 *
   -0.4059
   -0.4059
```

说明 从上面的结果中可以看出，由于示例中的二元函数是平滑的，因此求解算法能够快速求解出数值结果。由于在上面的代码中设置了显示迭代，在此将显示有关迭代中的优化信息。

3.6 数值积分

微积分是高等数学中的重要知识，在工程实践中，微积分有着十分广泛的应用。[illegible]计算其数值积分是十分重要的内容。在MATLAB中，用户可以使用多种方法来求解数值积分，包括数值积分、符号积分和Simulink模拟积分等。在本节中，将主要介绍数值积分的方法，并辅以介绍符号积分和Simulink积分等方法。

3.6.1. 一元函数的数值积分

[illegible]

- [illegible]
- [illegible]

下面详细介绍上面函数的参数含义。

- fun：被积函数，可以是字符串、匿名函数、内联函数、M 文件的函数句柄，被积函数中一般使用 x 来作为自变量。
- a、b：被积函数的上限和下限，必须都是确定的数值。
- tol：可选，控制绝对误差，默认的数值精度是 10^{-6}。
- trace：[illegible]

[illegible]

例 3.31 求解 $\int_{0}^{\pi}(4\cos(2t)^2+\sin(t)+1)dt$ 的数值。

Step 1 [illegible]，这个数值求解是基于曲线参数方程，函数对应的参数方程如下

$$\begin{cases} x(t)=\sin(2t) \\ y(t)=\cos(t) \\ z(t)=t \end{cases}$$

Step 2 [illegible]，在 MATLAB 的命令窗口中输入下面的程序代码

```
% 绘制函数曲线
[illegible]
[illegible]
[illegible]
grid on
```

Step 3 [illegible]

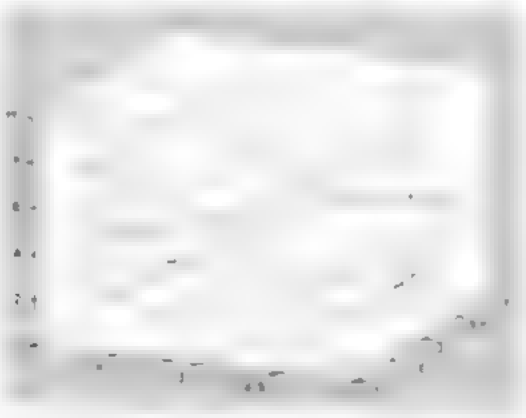


图 3.10 函数图形

Step 4 [illegible]

开M文件编辑器，在其中输入下面的程序代码

```
%编写被积函数的M文件
function f = hcurve(t)
f = sqrt(4*cos(2*t).^2 + sin(t).^2 + 1);
```

今输入上面的程序代码后，将代码保存为“hcurve.m”文件。

Step 5 [illegible]

```
>> len1=quad(@hcurve,0,3*pi);
>> len2=quadl(@hcurve,0,3*pi);
```

Step 6 [illegible]

```
len1 =
   17.2220
len2 =
[illegible]
```

Step 7 [illegible]

```
[illegible]
[illegible]
```

Step 8 [illegible]

```
        9      0.0000000000     2.55458120e+000     4.5151652105
       11      0.0000000000     1.27979060e+000     2.1975964146
       13      0.0000000000     6.39895300e-001     1.1908151623
       15      0.6398452996     6.39895300e-001     0.9939908843
                 ..............................//限于篇幅，省略了部分数据
       53      0.1443673615     1.27979060e+000     2.1975964146
       55      0.1449873615     6.39895300e-001     0.9939908843
       57      0.7848826611     6.39895300e-001     1.1908151623
len3 =
[illegible]
       18      0.0000000000     4.71238898e+000    15.8755795718
       23      0.0000000000     4.32369745e-001     1.4707654142
       28      0.0000000000     3.96706633e-002     0.1768555623
       33      0.0793413265     7.98333951e-002     0.3426629563
       ..........................................//限于篇幅，省略了部分数据
      208      0.6393797974     7.98333951e-002     0.1969870400
      213      0.7990465876     9.66608141e-002     0.2884072894
      218      0.9924082158     9.66608141e-002     0.3638528604
      223      1.1857698440     7.98333951e-002     0.3426629563
      228      1.3454366343     3.96706633e-002     0.1768555623
len4 =
   17.2220
```

从上面文件运行结果，[illegible]

3.6.2 使用 Simulink 求解数值积分

例 3.32 使用 Simulink 求解 $\int_{0}^{10} 4\cos(2t) + \sin(t) + 1\,dt$ 数值

Step 1 选择 MATLAB 编辑器工具栏"[illegible]"[illegible]"[illegible]"命令，打开模型编辑器，[illegible]加相应的模型块。如图 3.11 所示。

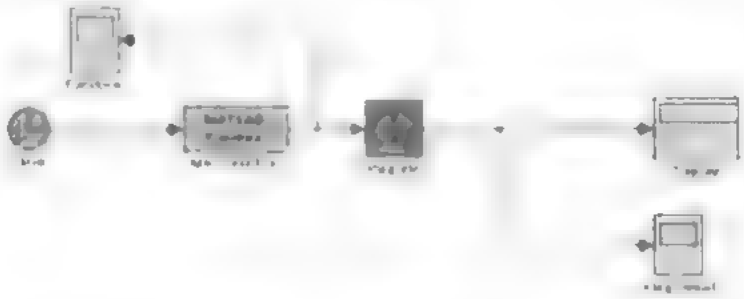


图 3.11 添加系统的模块

Step 2 在系统模块属性对话框中，[illegible]在系统模块中"[illegible]"模块，[illegible]框，在其中设置被积函数表达式。如图 3.12 所示。

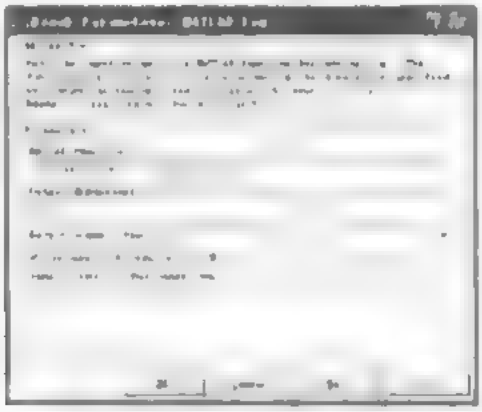


图 3.12 设置系统模块的属性

Step 3 设置系统仿真参数，[illegible]为 3π，然后运行仿真，保存的结果如图 3.13 所示。

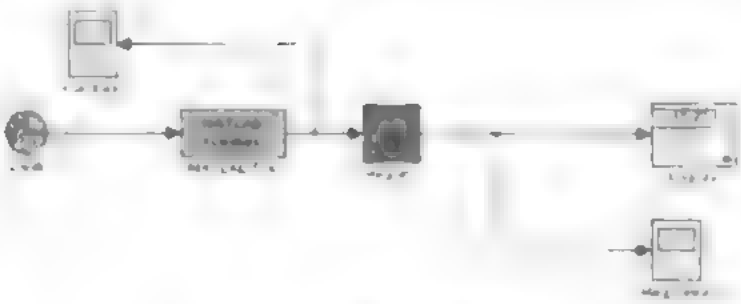


图 3.13 查看仿真的结果

Step 4 查看被积函数的选项，从工作空间中[illegible]，为了 Simulink 积分模型的结果保存在 17.22

同时，用户可以查看被积函数的图形，如图 3.14 所示。

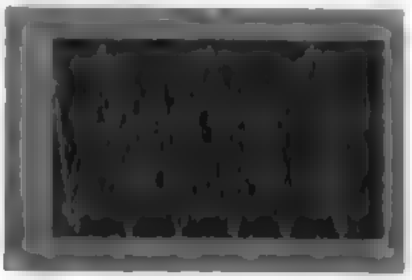


图 3.14 被积函数的图形

[illegible]

3.6.3 求解瑕积分

例 3.33 求解积分 $\int_0^1 \sqrt{\ln \frac{1}{x}}\,dx$ 的数值。

step 1 分析积分函数。在下面的步骤中表达式中，由于在 x=0 时，被积函数 $\sqrt{\ln \frac{1}{x}}$ 的数值趋近于无穷大，也就是 $\lim_{x\to 0} \sqrt{\ln \frac{1}{x}} \to \infty$。因此，上面的积分属于瑕积分或者称为反常积分，这和上面的有限积分在性质上有比较大的差异。此时根据基础的高等数学知识，该积分数值为 $\int_0^1 \sqrt{\ln \frac{1}{x}}\,dx = \frac{\sqrt{\pi}}{2} = 0.8862$。

step 2 求解积分数值。在 MATLAB 的命令窗口中输入下面的代码：

```
>>f=inline('sqrt(log(1/x))','x');
>> Infq=quad(f,0,1);
>> Infql=quadl(f,0,1);
```

step 3 查看求解结果。在命令窗口中输入变量名称，得到求解的结果如下：

```
Warning: Divide by zero.
> In inlineeval at 13
  In inline.subsref at 25
  In quad at 62
Infq =
    0.8862
Warning: Divide by zero.
> In inlineeval at 13
  In inline.feval at 34
  In quadl at 64
Infql =
    0.8862
```

step 4 使用符号积分求解。在 MATLAB 的命令窗口中输入下面的代码：

```
>> f=inline('sqrt(log(1/x))','x');
>> syms x
>> Is=vpa(int('sqrt(log(1/x))','x',0,1))
```

step 5 [illegible]

```
Warning: Explicit integral could not be found.
> In sym.int at 58
  In char.int at 9
 ls =
.0862269254527580136490837416705 7
```

[illegible]

3.6.4 矩形区域的多重数值积分

[illegible]中将主要介绍如何在 MATLAB 中计算二重数值积分。

[illegible] MATLAB，[illegible]dblquad，[illegible]

```
q = dblquad(fun,xmin,xmax,ymin,ymax,tol,method)
```

[illegible]

例 3.34 求解积分 $\int_0^{\pi}\int_{\pi}^{2\pi}(y\sin x + x\cos y)\,dx\,dy$ 的数值。

step 1 求解积分数值，在 MATLAB 命令窗口中，输入下面的语句：

```
>>integrand=@(x,y) y*sin(x)+x*cos(y);
>>xmin = pi;
>>xmax = 2*pi;
>>ymin = 0;
[illegible]
[illegible] dblquad(integrand,xmin,xmax,ymin,ymax)
```

step 2 在命令窗口中，输入上面的语句，按 "Enter" 键，即可求解积分数值。

```
result =
   -9.8694
```

step 3 使用符号运算求解积分数值。

```
[illegible]
>> result1=vpa(int(int((y*sin(x)+x*cos(y)),x,pi,2*pi),y,0,pi))
```

step 4 在命令窗口中，输入上面的语句，按 "Enter" 键，即可求解积分数值。

```
result1 =
 -9.8696044010893586188344909998761
```

变量区域的多重数值积分

前面所介绍的内容中，都是固定数值的二重积分运算方法，但是在实际应用中，二重积分并不都是矩形计算区域，在计算区域中会包含变量表达式。也就是说，积分区域可以表示成下面的情况：

$$R=\{(x,y)\,|\,a\leqslant x\leqslant b,c(x)\leqslant y\leqslant d(x)\}$$

用户需要求解的积分表达式为：

$$I=\int\int_R f(x,y)\mathrm{d}x\mathrm{d}y=\int_a^b\left\{\int_{c(x)}^{d(x)}f(x,y)\mathrm{d}y\right\}\mathrm{d}x$$

对于上面的积分表达式，进行数值计算的表达式为：

$$I(a,b,c(x),d(x))=\sum_{m=1}^{M}w_m\sum_{n=1}^{N}v_n f(x_m,y_{m,n})$$

在上面的表达式中 w_m、v_n 表示的是权重，取决于一维积分方法。关于上面二重积分的数值分析的方法，如图 3.15 所示。

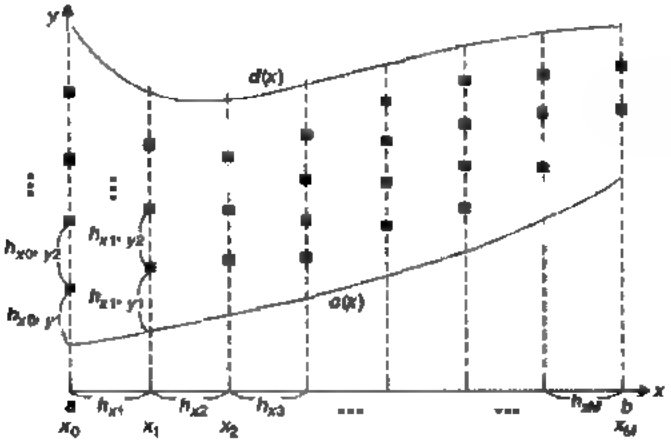


图 3.15 二重积分数据点

根据上面的数据点区域，用户需要自行编写M文件来计算上面的数值积分。在本小节中，将使用一个简单的例子说明如何计算二重数值积分。

例 3.35 求解积分 $I=\int_{-1}^{1}\int_0^{\sqrt{1-x^2}}\sqrt{1-x^2-y^2}\,\mathrm{d}y\mathrm{d}x$ 的数值。

step 1 编写一维数值积分的M文件。选择命令窗口编辑栏中的"File" ⇨ "New" ⇨ "M-File"命令，打开M文件编辑器，在其中输入下面的程序代码：

```
function INTf=smpsns_fxy(f,x,c,d,N)
%函数f(x,y)的一维数值积分数值;
%对应的积分区域是 Ry={c<=y<=d}

%当用户没有输入函数中的N参数时，默认值为100
if nargin<5
    N=100;
end

%当参数c=d或者参数N=0时，返回积分数值为0
if abs(d-c)<eps|N<=0
    INTf=0;
```

```
    return;
end
%如果参数N是奇数，则将其加1，变成偶数
if mod(N,2)~=0
    N=N+1;
end
%计算单位高度数值
h=(d-c)/N;
%计算节点的y轴坐标值
y=c+[0:N]*h;
%计算节点的积分函数数值
fxy=feval(f,x,y);
%确定积分的限制范围
fxy(find(fxy==inf))=realmax;
fxy(find(fxy==-inf))=-realmax;
%计算奇数和偶数的节点x坐标数值
kodd=2:2:N;
keven=3:2:N-1;
%根据积分公式得出积分数值
INTf=h/3*(fxy(1)+fxy(N+1)+4*sum(fxy(kodd))+2*sum(fxy(keven)));
```

在输入上面的程序代码后，将代码保存为“smpsns_fxy.m”文件。

step 2 编写二重数值积分的M文件。选择命令窗口编辑栏中的“File”⇨“New”⇨“M File”命令，打开M文件编辑器，在其中输入下面的程序代码：

```
function INTfxy=int2s(f,a,b,c,d,M,N)
% 被积函数f(x,y)的二重积分数值
% 积分区域为R ={(x,y)|a<=x<=b,c(x)<=y<=d(x)}
% 使用的积分方法是Simpson法则

if ceil(M)~=floor(M)
    hx=M;
    M=ceil((b a)/hx);
end
if mod(M,2)~=0
    M=M+1;
end
hx=(b-a)/M;
m=1:M+1;
x=a+(m-1)*hx;

%判断参数c是否是数值
%如果c是数值，将积分限制设置为数值c
%如果c不是数值，则将积分显示设置为函数表达式
if isnumeric(c)
    cx(m)=c;
else
    cx(m)=feval(c,x(m));
end

%判断参数d是否是数值
%如果d是数值，将积分限制设置为数值c
%如果d不是数值，则将积分显示设置为积分表达式
if isnumeric(d)
    dx(m)=d;
```

```
else
    dx(m)=feval(d,x(m));
end

%重复和参数M类似的操作
if ceil(N)~=floor(N)
    hy=N;
    Nx(m)=ceil((dx(m)-cx(m))/hy);
    ind=find(mod(Nx(m),2)~=0);
    Nx(ind)=Nx(ind)+1;
else
    if mod(N,2)~=0
        N=N+1;
    end
    Nx(m)=N;
end

%根据Simpson法则计算各个节点的数值
for m=1:M+1
    sx(m)=smpsns fxy(f,x(m),cx(m),dx(m),Nx(m));
end

%计算奇数和偶数的节点
kodd=2:2:M;
keven=3:2:M-1;
%计算积分数值
INTfxy=hx/3*(sx(1)+sx(M+1)+4*sum(sx(kodd))+2*sum(sx(keven)));
```

在输入上面的程序代码后，将代码保存为“int2s.m”文件。

step 3 进行二重积分计算。在 MATLAB 的命令窗口中输入下面的代码：

```
>> x=[-1:0.05:1];
>> y=[0:0.05:1];
>>[X,Y]=meshgrid(x,y);
>>f510=inline('sqrt(max(1-x.*x y.*y,0))','x','y');
>> Z=f510(X,Y);
>> d=inline('sqrt(max(1-x.*x,0))','x');
>> b=1;
>> a=-1;
>> c=0;
>> Vs1=int2s(f510,a,b,c,d,100,100);
>> error1=Vs1-pi/3;
>> Vs2=int2s(f510,a,b,c,d,0.01,0.01);
>> error2=Vs2-pi/3;
```

step 4 查看求解结果。在命令窗口中输入变量名称，然后按“Enter”键，得到求解的结果如下：

```
>> Vs1
Vs1 =
    1.0470
>> Vs2
Vs2 =
    1.0470
>> error1
error1 =
```

```
-1.5315e-004
>> error2
error2 =
-1.9485e-004
```

在上述的程序结果中，V-1 和 V-2 是分别采用了不同的计算方法得到的结果，error1 和 error2 是计算结果和真实结果之间的误差。

step 5 绘制函数图形。在 MATLAB 的命令窗口中输入下面的代码

```
>> surf(X,Y,Z)
>>shading interp
>> colormap hsv
>> colorbar horiz
```

step 6 查看结果。输入上面的程序代码后，按"Enter"键，得到的结果如图 3.16 所示。

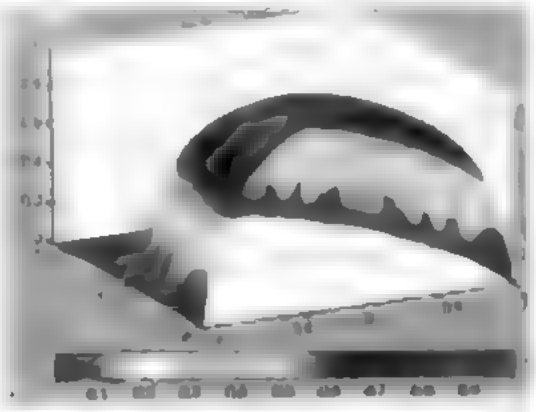


图 3.16 函数图形

3.7 概率论和数理统计

本节将主要介绍在 MATLAB 中应用概率论和数理统计的方法，主要的内容包括概率分布、数理统计和假设检验等。在每个具体的小节中，将主要介绍如何在 MATLAB 中运用相关知识，对于具体的背景知识，请读者查看对应的书籍。

3.7.1 双变量的概率分布

概率分布是概率论和数理统计的基础知识。在 MATLAB 中，提供处理常见概率分布的各种命令，包括二项分布、泊松分布、正态分布、t 分布等概率分布。这些内容比较简单，在本小节就不详细展开介绍了，感兴趣的读者可以查阅相应的帮助文件。

在本小节中，将主要介绍如何在 MATLAB 中处理两变量或者多变量的概率分布的情况。首先介绍如何处理双变量 t 分布（bivariate t distribution），根据单变量的概率知识，描述双变量 t 分布的参数主要有数据矩阵、相关系数矩阵和自由度 ν。下面举例说明如何在 MATLAB 中处理多元分布的问题。

例 3.36 在 MATLAB 中使用直方图来显示双变量 t 分布，其中两个变量服从的分布分别为 N(1,n5)，N(0,n5)，其中前者，两个变量的自由度分别为 1 和 5。下面给出使用程序来显示两个变量线性相关的情况

的不同取值下的分布情况。

step 1 绘制二元概率分布的图形。在 MATLAB 的命令窗口中输入下面的代码：

```
%设置分布参数
%n 代表数据点个数
%nu 表示自由度
%相关系数矩阵为[ 1 .8; .8 1]
n = 500;
nu = 1;
%产生多元 t 分布的随机数值矩阵
T = mvtrnd([ 1 .8; .8 1], nu, n);
%计算 t 分布数值的累积概率分布数值
U = tcdf(T,nu);

%绘制数据点的图形，并设置图形的属性
subplot(2,2,1);
plot(U(:,1),U(:,2),'r.');
title('rho = 0.8');
xlabel('U1');
ylabel('U2')

%相关系数矩阵为[ 1 .1; .1 1]
T = mvtrnd([ 1 .1; .1 1], nu, n);
U = tcdf(T,nu);
%绘制数据点的图形，并设置图形的属性
subplot(2,2,2);
plot(U(:,1),U(:,2),'r.');
title('rho = 0.1');
xlabel('U1');
ylabel('U2');

%相关系数矩阵为[ 1 -.1; -.1 1]
T = mvtrnd([ 1 -.1; -.1 1], nu, n);
U = tcdf(T,nu);
%绘制数据点的图形，并设置图形的属性
subplot(2,2,3);
plot(U(:,1),U(:,2),'r.');
title('rho = -0.1');
xlabel('U1');
ylabel('U2');

%相关系数矩阵为[ 1 -.8; -.8 1]
T = mvtrnd([ 1 -.8; -.8 1], nu, n);
U = tcdf(T,nu);
%绘制数据点的图形，并设置图形的属性
subplot(2,2,4);
plot(U(:,1),U(:,2),'r.');
title('rho =  0.8');
xlabel('U1');
ylabel('U2')
```

step 1 查看图形。在输入程序代码后，按“Enter”键，得到的图形如图 3.17 所示。

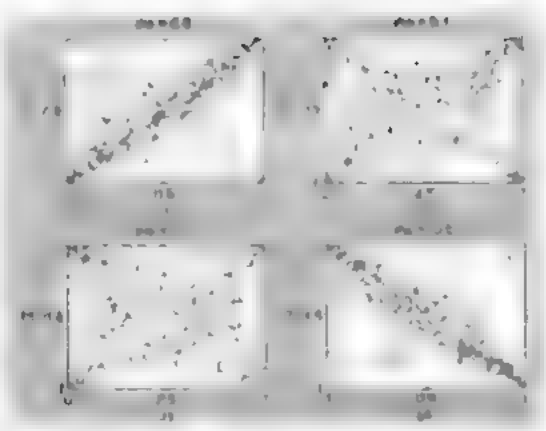


图 3.17　双变量 t 分布的图形

关于上面的程序代码，相关说明如下

- 在上面的程序代码中，mvtrnd 命令的功能是从多元 t 分布中产生随机数据矩阵，关于其具体的用法，读者可以查看对应的帮助文件。
- tcdf 命令的功能是产生 t 分布的累积概率数值，具体的用法请查看相应的帮助文件。

提示

从上面的图形中可以看出，当两个变量不相关时，其概率密度值随着横坐标的增大而减小，两个变量的分布呈如椭圆形；而当两个变量之间的相关系数为正数值时，相关值越大时，两个变量的分布呈如椭圆

3.7.2 不同概率分布

在 MATLAB 中，除了绘制两个相同分布变量之外，还可以绘制两个不同随机分布的变量的数据分布图，下面举例详细说明。

例 3.37　两个相关随机变量，分别服从 Gamma 分布和 t 分布，两个变量相互独立，且具体的随机变量参数为 Gamma(2,1)和 t(5)，在 MATLAB 中绘制两个变量的数据分布图形。

step 1 绘制二元概率分布的图形。在 MATLAB 的命令窗口中输入下面的代码

```
subplot(1,1,1);

%设置概率分布的参数
n = 1000;
rho = .7;
nu = 5;

%产生多元t分布的随机数值矩阵
T = mvtrnd([ 1 rho; rho 1], nu, n);
%计算t分布数值的累积概率分布数值
U = tcdf(T,nu);
%产生两个概率分布的数值
X = [gaminv(U(:,1),2,1) tinv(U(:,2),5)];

%计算两个直方图的数值
[n1,ctr1] = hist(X(:,1),20);
[n2,ctr2] = hist(X(:,2),20);
```

```
%绘制图形
subplot(2,2,2);
[illegible],'.'),
axis([0 15 -10 10]);
h1 = gca;
title('1000 Simulated Dependent t and Gamma Values');
xlabel('X1 ~ Gamma(2,1)');
ylabel('X2 ~ t(5)');
subplot(2,2,4); bar(ctr1,-n1,1);
axis([0 15 -max(n1)*1.1 0]);
axis('off');
h2 = gca;
[illegible]
axis([ -max(n2)*1.1 0 -10 10]);
[illegible],
h3 = gca;
set(h1,'Position',[0.35 0.35 0.55 0.55]);
set(h2,'Position',[ .35 .1 .55 .15]);
set(h3,'Position',[ .1 .35 .15 .55]);
colormap([ .8 .8 1]);
```

step 2 [illegible]

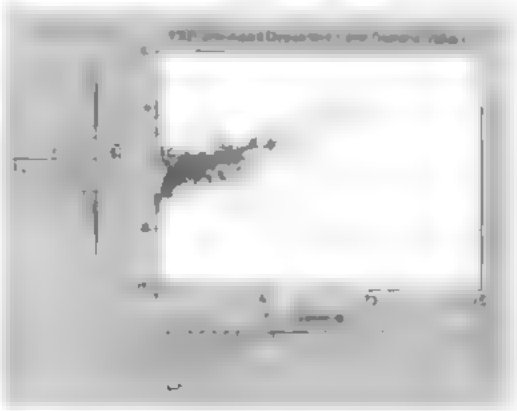


图 3.18 两个独立随机变量的图形

3.7.3 数据分布分析

[illegible]

例 3.38 [illegible] 变量[illegible]分布 $N(\mu_1,\mu_2,\sigma_1,\sigma_2)$，其中参数[illegible]和推断是否正确。

step 1 [illegible]

```
%产生随机数据
x = [trnd(20,1,50) trnd(4,1,100)+3];
%设置混合概率密度函数
[illegible]
                    p*normpdf(x,mu1,sigma1) + (1-p)*normpdf(x,mu2,sigma2);
```

```
%设置参数的数值
pStart = .5;
muStart = quantile(x,[ .25 .75]);
sigmaStart = sqrt(var(x) - .25*diff(muStart).^2);
start = [pStart muStart sigmaStart sigmaStart];
%设置参数的上下限
lb = [ 0 -Inf -Inf 0 0];
ub = [ 1 Inf Inf Inf Inf];
%设置求解的属性
options = statset('MaxIter',300, 'MaxFunEvals',600);
[phat,pci] = mle(x, 'pdf',[illegible], 'start',start, ...
                                'lower',lb, 'upper',ub, 'options',options);
%绘制原始数据的直方图
bins = -2.5:.5:7.5;
h = bar(bins,histc(x,bins)/(length(x)*.5),'histc');
set(h,'FaceColor',[ .9 .9 .9]);
[illegible]
%绘制概率密度曲线
[illegible]
hold on; plot(xgrid,pdfgrid,'-');
[illegible]
xlabel('x'); ylabel('Probability density');
```

Step 2 单击菜单栏中的"输入并保存文件"，按"Enter"键，则会绘制出如图 3.19 所示的图形。

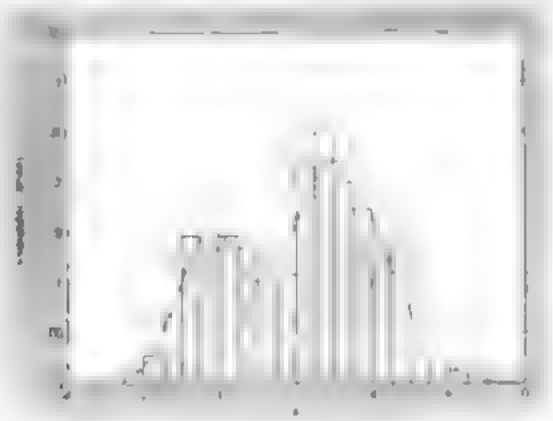


图 3.19 数据分布图形

3.7.4 假设检验

假设检验是数理统计中的一个重要内容，在 MATLAB 软件中，[illegible]，例如，在某个指定产品，每日平均产量、[illegible]、均值检验、[illegible]，在 MATLAB 来进行假设检验。

例 3.39 计算正态数据的一个[illegible]，已知总体标准差为 1，并可以设检验的置信水平为 5%，假设检验的平均值为 100。

Step 1 进行假设检验，为 MATLAB 的[illegible]，输入[illegible]

```
%设置假设检验的参数
mu0 = 100;
```

```
sig = 20;
N = 16;
%设置假设检验的置信水平
alpha = 0.05;
conf = 1-alpha;
%设置正态分布的截断点
cutoff = norminv(conf, mu0, sig/sqrt(N));
%产生数据点
x = [ linspace(90,cutoff), linspace(cutoff,127)];
y = normpdf(x,mu0,sig/sqrt(N));
%绘制正态分布图形
h1 = plot(x,y);
xhi = [ cutoff, x(x>=cutoff)];
yhi = [ 0, y(x>=cutoff)];
%绘制假设检验的面积图
patch(xhi,yhi,'b');
%设置图形的标题和坐标轴名称
title('Distribution of sample mean, N=16');
xlabel('Sample mean');
ylabel('Density');
text(cutoff,0.0,sprintf('Reject if mean>%.1f',cutoff),'Color','r',
'Ha','
```

Step 2 在命令窗口中输入程序代码后，按“Enter”键，程序运行结束后，[illegible]

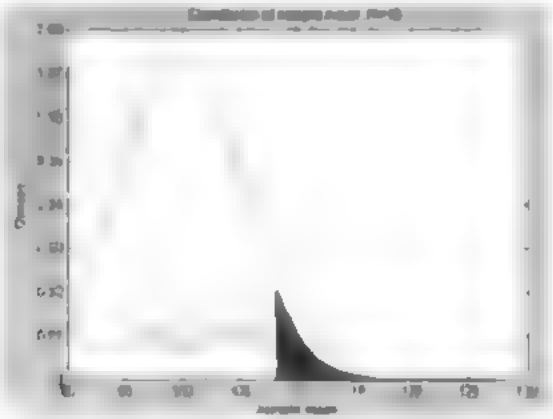


图 3.20 单侧假设检验图形

提示 在上面的程序代码中，首先输入关于原假设检验的参数，然后根据关于检验统计量的抽样分布的信息，最后在上个步骤的曲线上标出临界值的位置。

Step 3 修改假设检验的参数，进行假设检验，修改假设检验的参数，在上面的基础上，重新进行假设检验。

在 MATLAB 的命令窗口中输入下面的代码

```
%修改假设检验的均值数值
mu1 = 110;
%计算正态分布数据点
y2 = normpdf(x,mu1,sig/sqrt(N));
%绘制正态分布曲线
```

```
h2 = line(x,y2,'Color','r');
%绘制假设检验的面积图
yhi = [ 0, y2(x>=cutoff)];
patch(xhi,yhi,'r','FaceAlpha',0.25);
%添加图形的提示信息
P = 1 - normcdf(cutoff,mu1,sig/sqrt(N));
text(lim,.7b,sprintf('Reject at P>%.4g Pbrob = %.2g',cutoff,P),'Color',
[1 0 0]);
legend([h1,h2],'Null hypothesis','Alternative hypothesis');
```

step 4 查看图形。在输入程序代码后，按"Enter"键，得到的图形如图3.21所示。

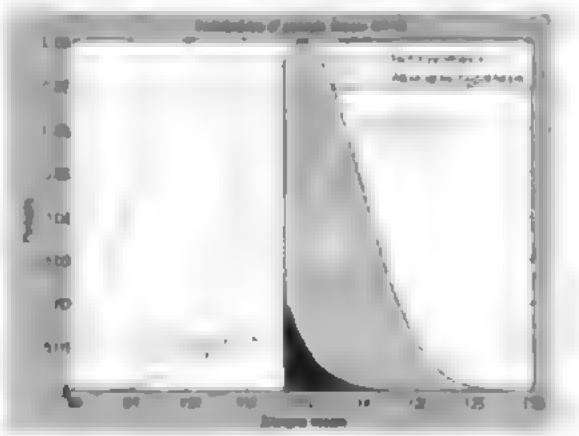


图3.21 修改条件后的假设检验

step 5 绘制累积概率密度函数的图形。为了对比了解两种不同的假设检验条件，由户可以绘制概率密度函数的图形。在MATLAB的命令窗口中输入下面的程序代码

```
%计算累积概率密度函数
ynull = normcdf(x,mu0,sig/sqrt(N));
yalt = normcdf(x,mu1,sig/sqrt(N));

%绘制累积密度函数图形
plot(x,ynull,'b-',x,yalt,'r-');

%计算置信条件水平下的反正态分布数值
zval = norminv(conf);
cutoff = mu0 + zval * sig / sqrt(N);

%绘制图形
line([ 90,cutoff,cutoff],[ conf, conf, 0],'LineStyle',':');
msg = sprintf(' Cutoff = %.2g = mu0 + %.2g*sigma / sqrt(N)',zval);
text(cutoff,.15,msg,'Rotation','b');
text(min(x),conf,sprintf('  %g%% test',100*alpha),'Color','b',...
                         'Vert'alignment','top',
Pal' = normcdf(cutoff,mu1,sig/sqrt(N));
line([ 90,cutoff],[ palt,palt],'Color','r','LineStyle',':');
text(90,palt+.02,sprintf(' Power = %.4g',palt),'Color',
[ 0 0 0]);
```

step 6 查看图形。在输入程序代码后，按"Enter"键，得到的图形如图3.22所示。

图 3.22 累积概率密度图形

[illegible]

step 7 绘制Power函数 [illegible]

```
% 定义需要的 Power 参数数值
DesiredPower = 0.80;
Nvec = 1:30;
cutoff = mu0 + norminv(conf)*sig./sqrt(Nvec);
% 计算假设检验的power数值
power = 1 - normcdf(cutoff, mu1, sig./sqrt(Nvec));
% 绘制图形
[illegible]
xlabel('N - sample size'); ylabel('Power')
title('Power function for the alternative: \mu = 110')
```

step 8 [illegible]

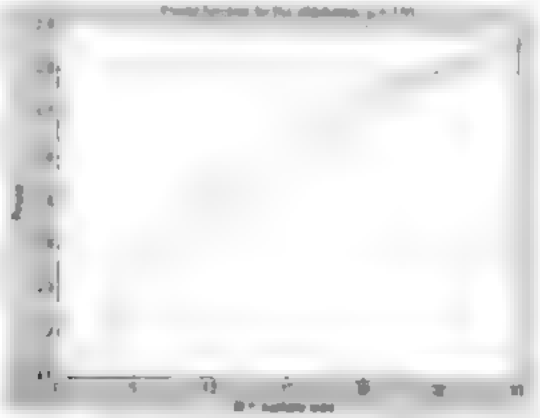


图 3.23 Power 函数的图形

step 9 使用Monte [illegible] 程序代码

```
% 定义Monte Carlo模拟的参数
```

```
%nsamples 表示样本数值
%smaplenum 是样本中数据点

nsamples = 400;
samplenum = 1:nsamples;
N=25;

% 创建零值矩阵
h0 = zeros(1,nsamples);
h1 = h0;

% 进行右侧已知方差条件下均值假设检验
for j=1:nsamples
    Z0 = normrnd(mu0,sig,N,1);
    h0(j) = ztest(Z0,mu0,sig,alpha,'right');
    Z1 = normrnd(mu1,sig,N,1);
    h1(j) = ztest(Z1,mu0,sig,alpha,'right');
end
p0 = cumsum(h0) ./ samplenum;
p1 = cumsum(h1) ./ samplenum;
% 绘制对应的图形
plot(samplenum,p0,'b-',samplenum,p1,'r-')
xlabel('Sample number'); ylabel('Proportion significant')
title('Verification of power computation')
legend('Null hypothesis','Alternative hypothesis','Location','East')
```

step 10 查看图形。在输入程序代码后，按"Enter"键，得到的图形如图 3.24 所示。

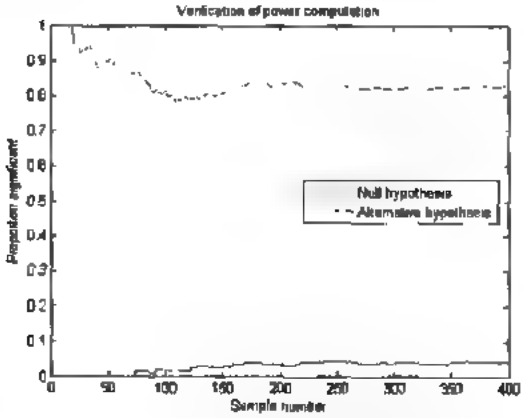


图 3.24　Monte carlo 检验图形

小结

在本章中，依次向读者介绍了矩阵分析、矩阵分解、特征值计算、数值积分、数理统计等内容，这些内容是 MATLAB 进行数值运算的重要部分，其中关于矩阵的分析和运算是其他操作的基础内容，希望用户能够熟练掌握。下一章，将介绍如何使用 MATLAB 进行数据分析。

✦　✦　✦

第4章 数据分析

本章包括

- 数据插值
- 傅里叶（Fourier）分析
- 常微分方程
- 曲线拟合
- 优化

数据分析和处理在各个领域有着广泛的应用，尤其是在数学、物理等科学领域和工程领域的实际应用中，会经常遇到进行数据分析的情况。例如，在工程领域根据有限的已知数据对未知数据进行推测时经常需要用到数据插值和拟合的方法，在信号工程领域则经常需要用到傅里叶变换的工具，在物理或工程领域中则经常需要根据现实情况抽象出微分方程，进行数值求解，等等。这些常见的数据分析方法在MATLAB中都可以实现，同时，MATLAB本身有着强大的数据分析和处理功能，还可以处理类似优化、偏微分方程等比较复杂的问题。

对于熟悉MATLAB基本命令的读者而言，通过本章的学习，可以了解到各命令的使用场合以及内在关系，同时获得综合运用命令解决实际问题的思路和借鉴的经验。本章将分别介绍插值、拟合、傅里叶变换、常微分方程、优化和偏微分方程等各种内容，从总体上来讲，本章各节之间没有依从关系，读者可以根据需要选择阅读相关章节，同时，本章中所列举的各个例子都是独立完整的，用户可以在自己的机器上运行。

插值

插值（Interpolation）是指在所给的基准数据情况下，研究如何平滑地估算出基准数据之间其他点的函数数值。每当其他点上函数值若获取的代价比较高时，插值就会发挥作用。

在MATLAB中提供多种插值函数，这些插值函数在获得数据的平滑度、时间复杂度和空间复杂度方面有着完全不同的性能。本节将主要介绍一维插值命令interp1，二维插值命令interp2等MATLAB内置函数。同时，还将根据不同的插值方法自行编写M文件，完成不同的插值运算，例如Lagrange插值、Newton插值等，下面分小节详细介绍。

一维插值命令

在MATLAB中，一维插值表示的是对一维函数$y=f(x)$进行插值。为了让读者更加形象地了解一维插值的含义，下面列出一维插值的图形，如图4.1所示。

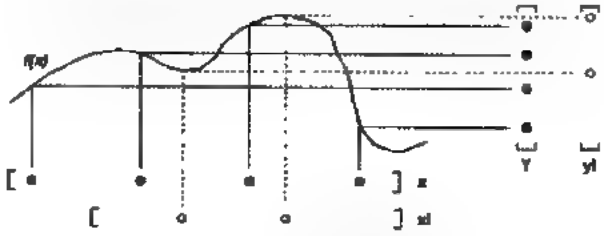


图4.1 一维插值的含义

[illegible]

值的位置，纵坐标 y 表示插值后运算的数值。

在 MATLAB 中，一维数据的插值可通过函数 interp1 来实现，其具体语法格式如下：

- y = interp1(x,y,xi)：在给定向量 x 和 y 的情况下，计算 x 的子集 xi 的插值结果 y。向量 x 和 y 必须是长度相同的向量，而 xi 可以是任意长度的向量。x 的元素必须是单调的，且 xi 的元素可以是任意顺序的。
- yi = interp1(y,xi)：函数默认情况下，x 被设为 1:length(y)，即 y 是一个等间隔的向量。
- yi = interp1(x,y,xi,method)：输入参数 method 用来指定插值的方法，其他可选值可参阅本章后面的内容中详细介绍。
- y = interp1(x,y,xi,method,'extrap')：对超出数据范围的数据进行外推插值。
- yi = interp1(x,y,xi,method,extrapval)：对超出数据范围的数据进行外推插值，extrapval 数值一般为 NaN 或者 0。
- pp = interp1(x,y,method,'pp')：以数值 pp 数据的形式，返回一个使用 method 指定方法的分段多项式 pp 的方法。

在 MATLAB 中，还提供 interp1q 命令作为 interp1 的快速命令，用于快速处理不规则的数据。interp1q 命令不支持外推，使用 interp1 命令可以。[illegible]

在上面的命令中，method 参数的取值和对应的含义如下：

- nearest：最近邻插值，[illegible]
- linear：线性插值，[illegible]
- spline：[illegible]
- pchip：分段三次埃尔米特插值，[illegible]
- cubic：三次卷积插值，[illegible]
- v5cubic：MATLAB5 中使用的三次多项式插值。

4.1.2 一维插值实例

- [illegible]

例 4.1 [illegible]

Step 1 某城市在 1[illegible] 年间的年末，每隔 [illegible] 年统计该城市常住人口数量，得到统计结果如下：

[illegible]
150.697、179.323、203.212、226.505、249.633

[illegible] 数据对年份进行了[illegible]，[illegible] 数量进行推测，分别使用不同的插值方法。

Step 2 选择菜单栏中的 "File"→"New"→"M-file" 命令，打开 M 文件编辑器，在其中输入下面的程序代码：

```
%已知数据
t = 1900:10:1990;
p = [ 75.995  91.972  105.711  123.203  131.669...
      150.697  179.323  203.212  226.505  249.633];
%使用不同的方法进行插值运算
x = 1900:0.01:1990;
[illegible]
yi_spline=interp1(t,p,x,'spline');
[illegible]
[illegible]
%绘制对比图形
subplot(2,1,1);
plot(t,p,'ko');hold on;
[illegible]
[illegible]
grid on;
title('Linear Vs Spline');
subplot(2,1,2);
plot(t,p,'ko');hold on;
plot(x,yi_cubic,'m','LineWidth',1.5);hold on
[illegible]
grid on;
title('Cubic Vs V5Cubic');
%创建新的绘图窗口
figure
[illegible]
plot(t,p,'ko');hold on;
plot(x,yi_nearest,'g','LineWidth',1.5);hold on
grid on;
title('Nearest Method')
```

当输入上面的代码后，将该代码保存为"intexa.m"文件。

step 3 在命令窗口中输入"intexa"，然后按"Enter"键，[illegible]，使用"[illegible]"语句[illegible]，[illegible]

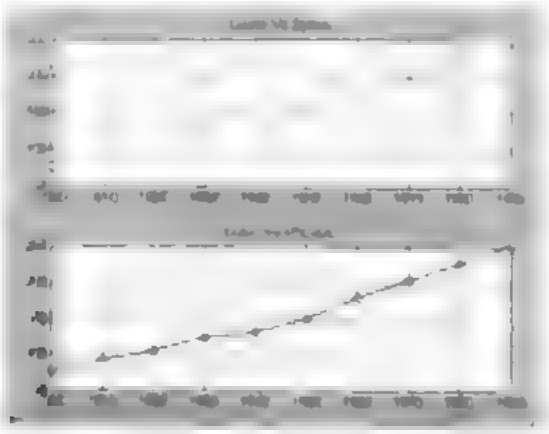


图 4.2 不同插值方法得到的结果

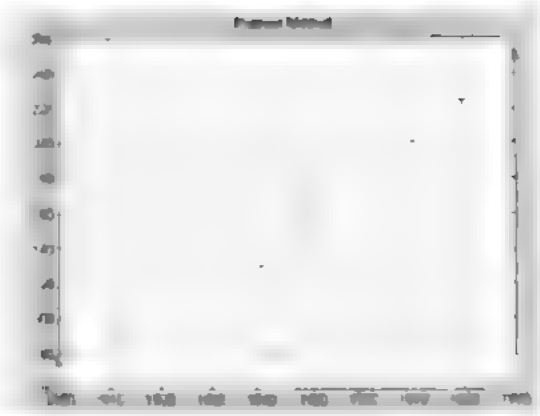


图4.3　使用"Nearest"方法得到的结果

step 4 [illegible]

```
msg='      year      Cubic       Linear       Nearest       Spline';
for i=0:8
n=10*i;
[illegible]
[illegible]
pop(i+1,2)=yi_cubic((year-1900)/0.01+1);
pop(i+1,3)=yi_linear((year-1900)/0.01+1);
[illegible]
[illegible]
end
[illegible]

disp(msg)
[illegible]
```

在输入上面的程序代码后，将该代码保存为"runexa.m"文件。

step 5 [illegible]'runexa'，[illegible]"Enter"键，得到的结果[illegible]

```
      year      Cubic        Linear       Nearest        Spline
      1905         84            84            92            85
      1915         99            99           106            98
      1925        115           114           123           115
      1935        127           127           132           128
      1945        140           141           151           139
      1955        165           165           179           165
      1965        192           191           203           192
      1975        215           215           227           215
      1985        238           238           250           238
      1985        238           238           250           238
```

说明　[illegible]"Nearest"[illegible]

[illegible]如表4.1所示。

表 4.1 各种插值方法的比较

方法	说明
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	进行速度最慢，精度高，最平滑

[illegible]

4.1.3 二维插值命令

[illegible]

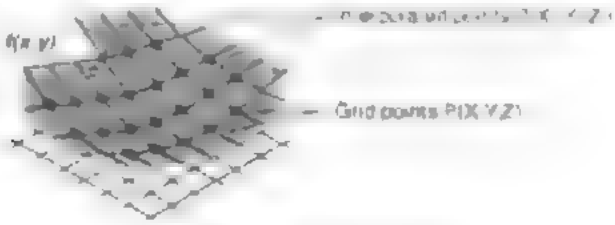


图 4.4 二维插值示意图

[illegible]

- [illegible] 函数 [illegible] 插值函数 [illegible] 点 [illegible] Z1 的(X1 Y1)在函数 $z=f(x, y)$ 上的数值
- [illegible] 是向量，长度为 $n \times m$，[illegible] $x=1:n$，$y=1:m$
- Z1 = interp2(Z,ntimes) 在两点之间通过内插进行插值 ntimes 次。
- [illegible] 插值。

[illegible] 参数，[illegible]

- [illegible] 使用模式与此类似。
- [illegible] interp1、interp2 是 [illegible] 函数，[illegible] interp3、interp4 [illegible] 增加，插值[illegible]，但[illegible]，[illegible] 情况相同，调用名称也很类似。
- 参数 [illegible] 是[illegible]，[illegible] 元素[illegible]，[illegible] 元素[illegible]，[illegible] 元素依照单调次序排列，但是任何两列都是相同的。
- 对于 [illegible] 数据系列，[illegible] 增量，[illegible] 数据系列。

关于这个命令的method参数，其实与之前在前面介绍的interp1命令中的method参数类似，只是参数数值'linear'表示采用双线性插值方法。

4.1.4 二维插值实例

例4.2 [illegible]函数$z(x,y)=\frac{\sin\sqrt{x^2+y^2}}{\sqrt{x^2+y^2}}$[illegible]，[illegible]"[illegible]"获取基础数据，生成一定基础坐标的"原始数据"，然后使用二维插值[illegible]要求的新数据，并绘[illegible]。

Step 1 选择菜单栏中的"File"→"New"→"M-file"命令，打开M文件编辑器，在其中输入下面的程序代码。

```
%生成基础原始数据
x=[illegible]:[illegible]pi;
y=x;
[X,Y]=meshgrid(x,y);
R=sqrt(X.^2+Y.^2)+eps;
[illegible]
[dzdx,dzdy]=gradient(Z);
dzdr=sqrt(dzdx.^2+dzdy.^2);
%绘制基础数据图形
surf(X,Y,Z,abs(dzdr))
colormap(spring)
[illegible]
[illegible]
```

完成上面的程序代码后，将程序代码保存为"intexp3.m"文件。

Step 2 单击[illegible]，在[illegible]中输入"intexp3"[illegible]，按"Enter"键，[illegible]

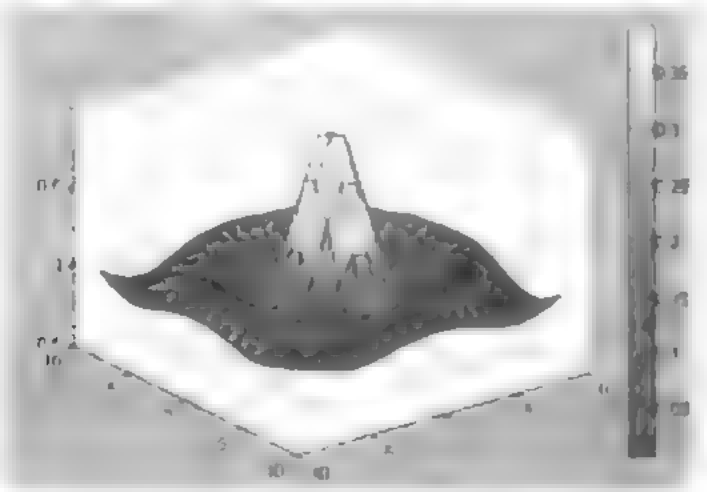


图4.5 根据基础数据绘制的三维图形

Step 3 求取二维插值的命令代码。打开上面步骤保存的"intexp3.m"文件，然后在M文件编辑器中添加下面的程序代码。

```
%进行二维插值运算
xi=linspace(-3*pi,3*pi,100);
[illegible]
```

```
[XI,YI]=meshgrid(xi,yi);
ZI=interp2(X,Y,Z,XI,YI,'cubic');
%绘制插值后的数据图形
figure
surf(XI,YI,ZI)
colormap(spring)
alpnamap('rampup')
colorbar
```

输入一些字符串，保存程序代码，单击工具栏中的 Debug 按钮，调试程序。

Step 6 单击程序编辑器工具栏中的 "Debug" 按钮，接着 "Exit" 键，程序运行结束，结果如图 4-6 所示。

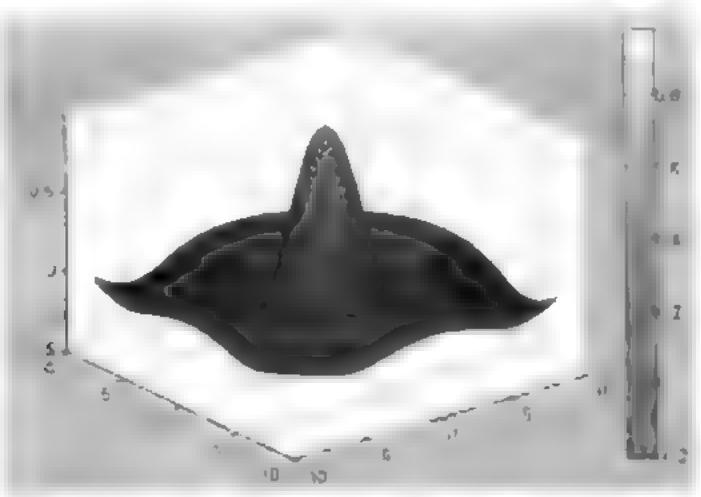


图 4-6 完成插值后的图形

例 4-3 在三维表面图上绘制等高线，使用三维流线图来表现数据的梯度，可以使用"streamline"命令在三维表面图上绘制数据的梯度。

Step 1 在 MATLAB 的命令窗口中输入下面的代码

```
%创建基础数据
>> z = peaks;
>>surf(z)
shading interp
hold on
%添加三维曲形表面的等高线
[c ch] = contour3(z,20); set(ch,'edgecolor','b')
%计算三维曲形表面的数值梯度
[u v] = gradient(z);
h = streamslice(-u,-v);
set(h,'color','k')
%对等高线进行二维数值插值
for i=1:length(h);
    zi = interp2(z,get(h(i),'xdata'),get(h(i),'ydata'));
    set(h(i),'zdata',zi);
end
view(30,50); axis tight
```

Step 2 查看程序结果，输入相关代码后，按 "Enter" 键，得到的结果如图 4-7 所示。

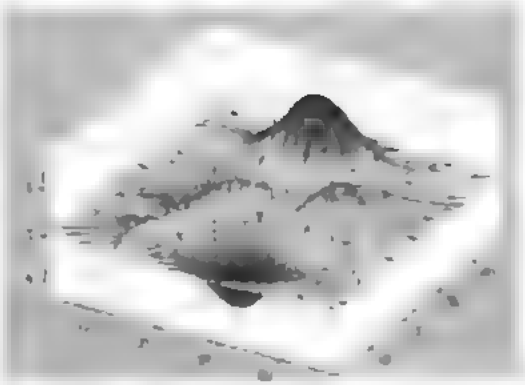


图4.7 完成的三维图形

[illegible]

4.1.5 样条插值

[illegible]，MATLAB还提供样条插值的方法。样条函数的主要特性是，[illegible]数据点，[illegible]多项式拟合的过程中，[illegible]样本数据，[illegible]多项式系数[illegible]。[illegible]，[illegible]，除[illegible]，由于样条[illegible]数据也能保证连续的一阶和二阶导数。

关于样条插值的主要命令如下

- yy = spline(x,y,xx)
- pp = spline(x,y)
- yy = ppval(pp,xx)

[illegible]，x,y表示[illegible]数据，xx[illegible]数据点[illegible]。[illegible]例子来说明Spline函数的使用方法。

例4.4 [illegible]，分别使用样条函数、[illegible]插值命令，并[illegible]插值结果进行比较。

step 1 在MATLAB命令窗口中输入下面的代码

```
>>x = -4:4;
>>y = [0 .15 1.12 2.36 2.36 1.46 .49 .06 0];
>>cs = spline(x,[0 y 0]);
>>xx = linspace(-4,4,101);
>>yyt = ppval(cs,xx);
>>yyt=interp1(x,y,xx,'spline');
>>plot(x,y,'o',xx,yyt,'-')
>>plot(xx,yyt,'m','LineWidth',1.5)
>>grid on
```

step 2 查看程序代码的结果。当输入上述代码后，按"Enter"键，得到的结果如图4.8所示。

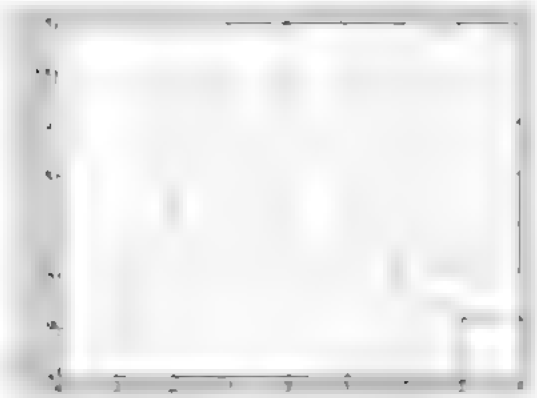


图4.8 显示插值图形

例4.5 使用样条插值进行圆形数据插值。

step 1 [illegible]

```
>> x = pi*[0:.5:2];
[illegible]
>>pp = spline(x,y);
[illegible]
>>grid on
>> axis equal
```

step 2 [illegible]

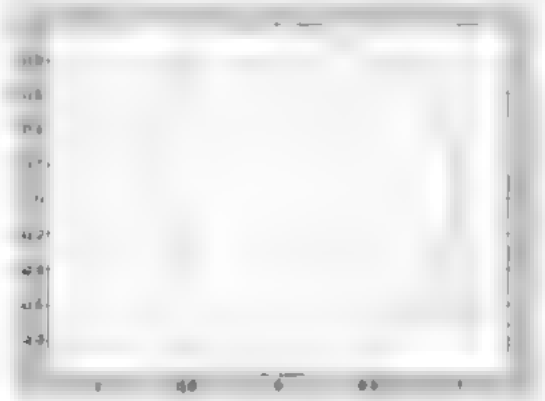


图4.9 圆形数据插值

4.1.6 牛顿插值原理

插值[illegible]牛顿[illegible]插值[illegible]算法[illegible]MATLAB[illegible]插值函数[illegible]插值[illegible]算法[illegible]newton[illegible]插值[illegible]数据插值。

当[illegible]插值[illegible]数据[illegible]$N+1$[illegible]数据[illegible]$\{(x_0,y_0),(x_1,y_1),\cdots,(x_N,y_N)\}$[illegible]$N-1$[illegible]满足[illegible]面的N个数据[illegible]$\{(x_0,y_0),(x_1,y_1),\cdots,(x_{N-1},y_{N-1})\}$的等式

$$n_N(x)=a_0+a_1(x-x_0)+a_2(x-x_0)(x-x_1)+$$
$$=n_{N-1}(x)+a_N(x-x_0)(x-x_1)\cdots(x-x_{N-1})$$

同时，需要满足的约束条件为 $n_0(x)=a_0$。

根据上面的关系式和推导，得到了牛顿插值多项式系数满足下面的关系式

$$a_0=y_0$$

$$a_1=\frac{y_1-a_0}{x_1-x_0}=\frac{y_1-y_0}{x_1-x_0}=Df_0$$

$$a_2=\frac{\frac{y_2-y_1}{x_2-x_1}-\frac{y_1-y_0}{x_1-x_0}}{x_2-x_0}=\frac{Df_1-Df_0}{x_2-x_0}=D^2f_0$$

由数学归纳法，得到牛顿插值方法的系数通式为

$$a_N=\frac{D^{N-1}f_1-D^{N-1}f_0}{x_N-x_0}=D^Nf_0$$

关于牛顿系数计算的过程，需要使用到一些基础的数学知识，有兴趣的，一探究竟吧，这里就不在这里详细介绍了。

4.17 牛顿插值实例

本小节给出一个简单的实例，说明如何使用牛顿插值方法进行数据插值。

例4.6 对函数 $f(x)=\frac{1}{1+8x^2}$，在等距节点，分别以 $n=5$ 和 $n=10$ 的牛顿插值多项式，比较不同的阶数的插值误差。

Step 1 选择菜单栏中的"File"→"New"→"M-file"命令，打开M文件编辑器，在其中输入下面的程序代码

```
function [n,DD]=newtonp(x,y)
% Newton 插值函数
% 输入参数  x=[x0 x1 ... xN]
%           y=[y0 y1 ... yN]
% 输出参数  n=Newton系数

N=length(x)-1;
DD=zeros(N+1,N+1);
DD(1:N+1,1)=y';
for k=2:N+1
    for m=k:N+2-k
        DD(m,k)=(DD(m+1,k-1)-DD(m,k-1))/(x(m+k-1)-x(m));
    end
end
a=DD(1,:);
n=a(N+1);
for k=N:-1:1
    n=n*(x-x(k))+a(k);
end
```

将上面的程序代码保存为"newtonp.m"文件，在后面的步骤中需要调用上面的程序代码进行牛顿插值运算。

Step 2 [illegible]

```
%计算4阶牛顿方法
x=[ -1 -0.5 0 0.5 1];
y=1./(1+8*x.^2);
n=newtonp(x,y);
xx=[ -1:0.02:1];
yy1=polyval(n,xx);
%计算原始函数的数据
[illegible]
%绘制图形
[illegible]
hold on;
%绘制插值图形
[illegible]
hold on;
%计算8阶牛顿方法
x1=[ -1:0.25:1];
[illegible]
n1=newtonp(x1,y1);
yy2=polyval(n1,xx);
%绘制新的插值图形
[illegible]
hold on;
[illegible]
hold on;
%计算10阶牛顿方法
x2=[ -1:0.2:1];
y2=1./(1+8*x2.^2);
n2=newtonp(x2,y2);
[illegible]
[illegible]
hold on;
[illegible]
grid on;
%添加图形的标题
[illegible]
hold off
```

Step 3 在运行[illegible]的结果，并输入一直线[illegible]，按“Enter”键，得到牛顿插值法结果，如图4.10所示。

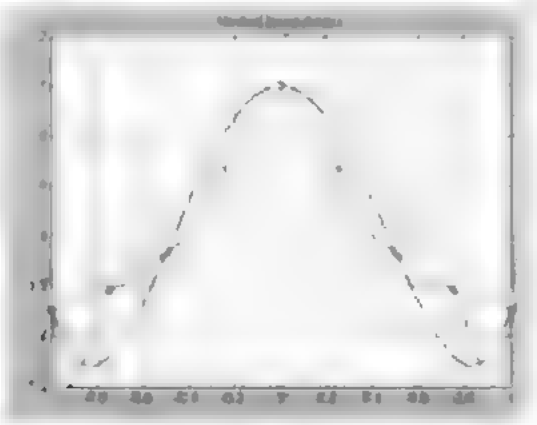


图4.10 牛顿插值图形

step 4 [illegible]

```
figure
plot(xx,yy1-yy,'b','LineWidth',1.5);hold on;
[illegible]
[illegible]
grid on
title('The error of Newton Interpolation')
```

step 5 [illegible]

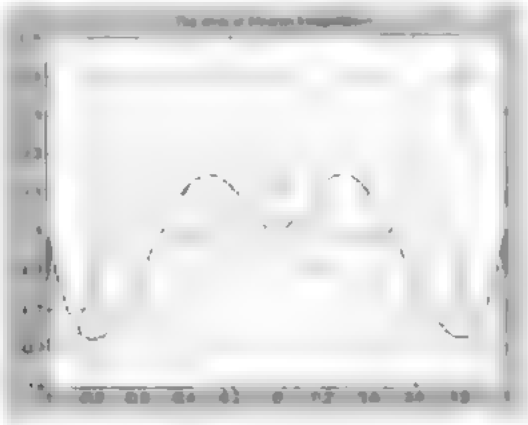


图 4.11 牛顿插值的误差结果

[illegible]

4.1.8 Chebyshev 多项式插值原理

[illegible] $f(x)=\frac{1}{1+bx}$ [illegible] 数据，然后使用基础数据进行整体数据的插值工作。

[illegible] 多项式提供下面的数据选择方法。

在标准化的数据范围[−1, +1]内选择下面的数据点

$$x_k=\cos\frac{2N+1-2k}{2(N+1)}\pi$$

[illegible] 关系

$$x_k=\frac{b-a}{2}x_k+\frac{a+b}{2}=\frac{b-a}{2}\cos\frac{2N+1-2k}{2(N+1)}\pi+\frac{a+b}{2}$$

[illegible]

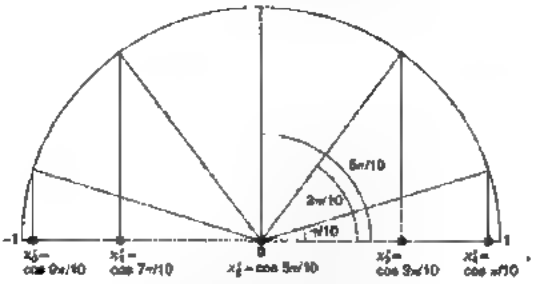


图 4.12 选择基础数据点

Chebyshev 多项式插值实例

例 4.7 以函数 $f(x)=\frac{1}{1+8x^2}$ 为基准函数，使用上面介绍的选择数据点的方法选择基础数据点，然后进行牛顿插值，最后比较不同的阶数的插值误差。

step 1 在 MATLAB 的命令窗口中输入下面的程序代码：

```
%计算4阶插值分析
N=4;
k=[0:N];
%选择基础数据点系列
x=cos((2*N+1-2*k)*pi/2/(N+1));
y=1./(1+8*x.^2);
c=newtonp(x,y);
xx=[-1:0.02:1];
yy=1./(1+8*xx.^2);
yy1=polyval(c,xx);
plot(xx,yy,'k-',x,y,'o');
hold on;
plot(xx,yy1,'b','LineWidth',1.5);
%计算8阶插值分析
N=8;
k=[0:N];
x=cos((2*N+1-2*k)*pi/2/(N+1));
y=1./(1+8*x.^2);
c1=newtonp(x,y);
plot(x,y,'*');
hold on;
plot(xx,yy2,'m','LineWidth',1.5);
%计算10阶插值分析
N=10;
k=[0:N];
x=cos((2*N+1-2*k)*pi/2/(N+1));
y=1./(1+8*x.^2);
c3=newtonp(x,y);
yy3=polyval(c3,xx);
plot(x,y,'d');
hold on;
plot(xx,yy3,'g','LineWidth',1.5);
grid on
Title('Chebyshev Interpolation')
```

step 2 [illegible]

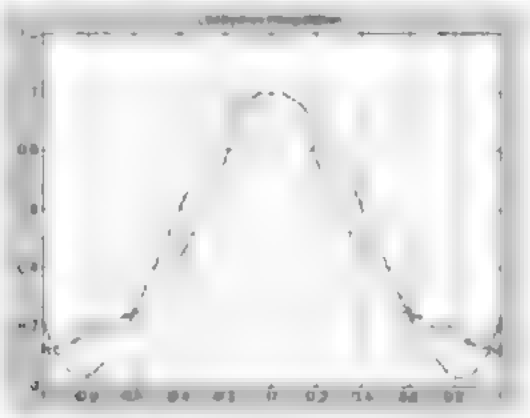


图 4.13　完成的插值结果

step 3 在 MATLAB 的命令窗口中输入下面的语句。

```
[illegible]
[illegible]
plot(xx,yy2-yy,'m','LineWidth',1.5);hold on;
[illegible]
grid on
title('The error of Chebyshev Interpolation')
legend('4 阶','8 阶','10 阶')
```

step 4 [illegible]

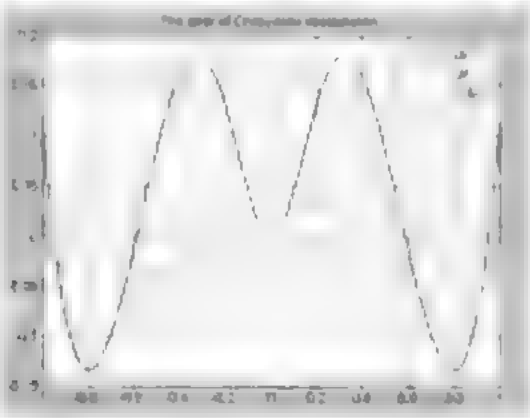


图 4.14　各阶插值误差

提示

[illegible]

4.2 曲线拟合

[illegible]

[illegible] y=f(x) [illegible] 合曲线拟合的方法。

[illegible] 用户同样要自行编写对应的M文件。

[illegible] 作方法。

4.2.1 多项式拟合

在MATLAB中，提供polyfit函数来[illegible]

- [p,S,mu] = polyfit(x,y,n)
- [y,delta] = polyval(p,x,S,mu)

其中，x和y表示[illegible]，参数μ表示[illegible]，其中 μ_1=mean(x)，μ_2=std(x)而且 $\mu=[\mu_1,\mu_2]$。

通过上面的命令，最后可以得到的拟合曲线的多项式为

$$y=p_1x^n+p_2x^{n-1}+\cdots+p_nx+p_{n+1}$$

[illegible]

例4.8 [illegible]

Step 1 [illegible]

```
[illegible]
>>y = erf(x);
%计算多项式拟合的参数
>> [p,s] = polyfit(x,y,6);
>> [yp,delta] = polyval(p,x,s);
[illegible]
x,yp-2*delta,'r:'), grid on
>> axis([0 5 0 1.4]);
>> Title('Polynomial curve fitting')
>> legend('[illegible]','Fitting')
```

Step 2 [illegible] 所示。

图4.15 多项式拟合结果

[illegible]

[illegible]

4.2.2 加权最小方差（WLS）拟合原理

[illegible]

$$A=\begin{vmatrix} x_1^n & \cdots & x_1 & 1 \\ x_2^n & \cdots & x_2 & 1 \\ \vdots & & \vdots & \vdots \\ x_m & & x_m & 1 \end{vmatrix},\quad \theta=\begin{vmatrix} \theta_n \\ \theta_{n-1} \\ \vdots \\ \theta \end{vmatrix}$$

使用加权最小方差（WLS）方法求解得到的拟合系数为

$$\hat{\theta}_0=\begin{vmatrix} \theta_n \\ \theta_{n-1} \\ \vdots \\ \theta \end{vmatrix}=(A^TWA)^{-1}A^TWy$$

[illegible]

[illegible]

4.2.3 加权最小方差（WLS）拟合实例

例4.9 [illegible]

WLS方法进行数据拟合。

step 1 选择命令窗口编辑栏中的“File”⇨“New”⇨“M-File”命令，打开M文件编辑器，在其中输入下面的程序代码：

```
function [th,err,yi]=polyfits(x,y,N,xi,r)
%x,y   :数据点系列
%N     :多项式拟合的系统
%r     :加权系数的逆矩阵

M=length(x);
x=x(:);
y=y(:);

%判断调用函数的格式
if nargin==4
%当用户调用函数格式为(x,y,N,r)
    if length(xi)==M
        r=xi;
        xi=x;
%当用户调用函数格式为(x,y,N, xi)
    else r=1;
    end
%当用户调用函数格式为(x,y,N)
elseif nargin==3
    xi=x;
    r=1;
end
%求解系数矩阵
A(:,N+1)=ones(M,1);
for n=N:-1:1
    A(:,n)=A(:,n+1).*x;
end

if length(r)==M
    for m=1:M
        A(m,:)=A(m,:)/r(m);
        y(m)=y(m)/r(m);
    end
end

%计算拟合系数
th=(A\y)';
ye=polyval(th,x);
err=norm(y-ye)/norm(y);
yi=polyval(th,xi);
```

在输入上面的代码后，将上面的代码保存为“polyfits.m”文件。

step 2 使用上面的程序代码，对基础数据进行LS多项式拟合。在MATLAB的命令窗口中输入下面的程序代码：

```
>>x=[-3.0 -2.0 -1.0 0 1.0 2.0 3.0]';
y=[-0.2774 0.8958 -1.5651 3.4565 3.0601 4.8568 3.8982]';
[x,i]=sort(x);
y=y(i);
```

```
x=[illegible] + [illegible]*(max(x)-min(x));
for i=1:4
N=[illegible] i;
[illegible]
subplot(220+i);
plot(x,y,'k*')
hold on
plot(xi,yi,'g:','LineWidth',1.5);
[illegible]
[illegible]
end
```

step 3 查看拟合过程，输入[illegible]，按[illegible]键，[illegible]如图4.16所示。

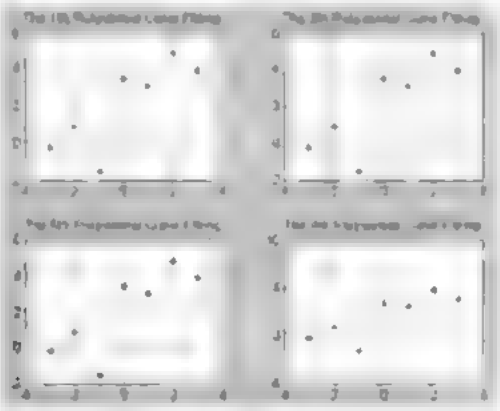


图4.16 使用LS方法求解的拟合结果

step 4 对上面的数据，[illegible]，[illegible]输入[illegible]

```
%基础数据
x=[-3.0 -2.0 -1.0 0 1.0 2.0 3.0]';
y=[-0.2774 0.8958 -1.5651 3.4565 3.0601 4.8568 3.6982]';
for i=1:4
N=[illegible] i;
[p,s]=polyfit(x,y,N);
[illegible]
[illegible]
plot(x,y,'k*')
hold on
[illegible]
title(['The ',num2str(N),'th Polynomial Curve Fitting'])
grid on
end
```

step 5 查看拟合的结果，输入[illegible]，按[illegible]键，[illegible]如图4.17所示。

从上面的例子中可看出，[illegible]，但对于给定的[illegible]基础数据，[illegible]，[illegible]，[illegible]自行体验比较。

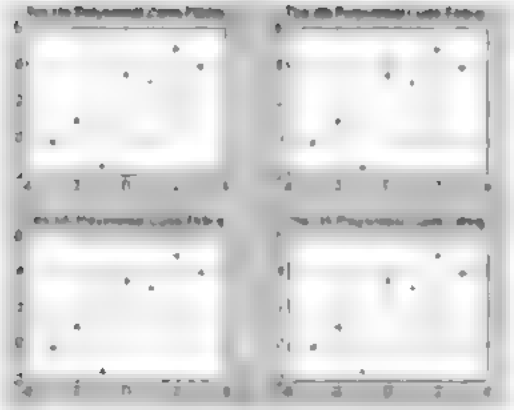


图 4.17 使用默认命令得到的拟合结果

例 4.10 [illegible]

Step 1 在 MATLAB 的命令窗口中输入下面的命令：

```
clear;
clc;
x=[1 3 5 7 9 2 4 6 8 10];
y=[0.0881 0.9290 2.4932 4.9292 7.9605 ...
    0.9536 2.4836 3.4173 6.3903 10.2443];
eb=[0.2*ones(5,1);ones(5,1)];
[x,i]=sort(x);
[illegible]
[illegible]
h=errorbar(x,y,eb,':');
[illegible]
hold on;
N=2;
[illegible]
[illegible]
[thwl,errwl,ywl]=polyfits(x,y,N,xi,eb);
plot(xi,yl,'g','LineWidth',1.5),
hold on
plot(xi,ywl,'r','LineWidth',1.5)
[illegible]
axis([0 10.5 -1 11.5]);
title('WLS VS LS');
legend([illegible],[illegible],'WLS')
box on
```

Step 2 单击[illegible]，一输入[illegible]按"Enter"键，查看拟合结果如图 4.18 所示。

从上面的图形结果中可以看出，[illegible]准确，[illegible]LS 的结果要好。

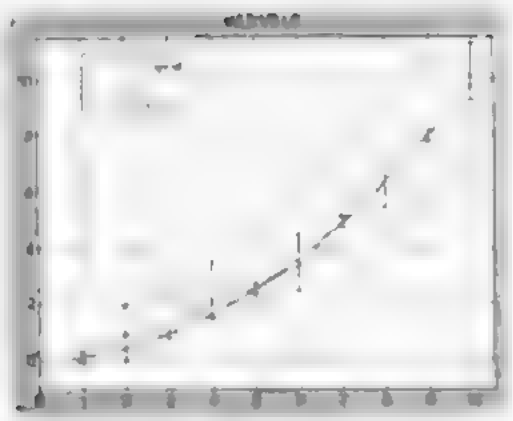


图 4.18　WLS 和 LS 方法得到的拟合结果

4.3　曲线拟合图形界面

在 MATLAB 中，为用户提供曲线拟合的界面，用户可以在交互式图形界面进行曲线拟合。在这种情况下，用户可以通过 spline 进行曲线拟合，用户可以对一组数据系，使用多项式拟合方法，绘制拟合曲线等多种功能。最后，还可以将拟合结果和数值保存到 MATLAB 的工作空间中。

4.3.1　曲线拟合

在本节中，将以一个简单的例子来说明如何使用该界面进行拟合工作。

例 4.11　[illegible]，使用曲线拟合工具箱完成拟合工作，下面将为读者详细介绍。

Step 1　在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> x=[ 1 3 5 7 9 2 4 6 8 10] ;
>>y=[ 0.0881 0.9290 2.4932 4.9292 7.9605 ...
     0.9536 2.4836 3.4173 6.3903 10.2443] ;
>>plot(x,y,'ro')
```

Step 2　[illegible]。输入代码后，按"Enter"键，得到图 4.19 [illegible]。

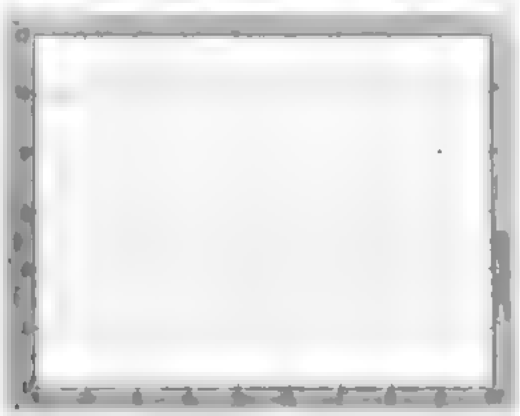


图 4.19　基础数据图形

step 3 [illegible]

打开关于函数拟合图形界面，如图 4.20 所示。

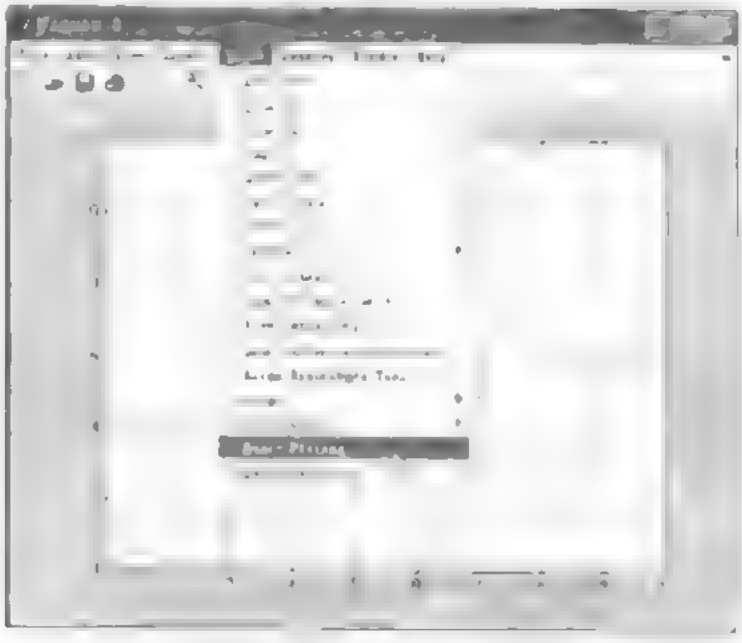


图 4.20 打开"Basic Fitting-1"对话框

step 4 [illegible]

图 4.21 默认的"Basic Fitting-1"对话框

step 5 [illegible]

如 4.22 所示。

[illegible]

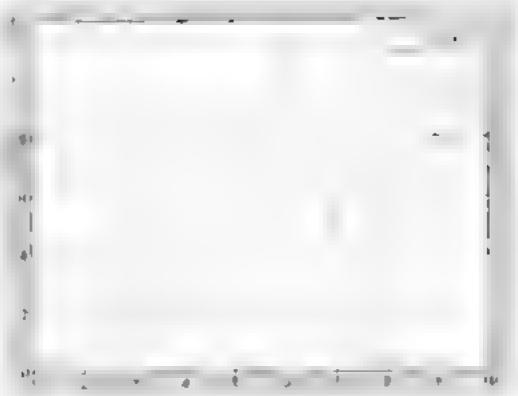


图 4.22　添加图例后的图形

STEP 6 [illegible]，然后选中“Show equation”选项，如图 4.23 所示。

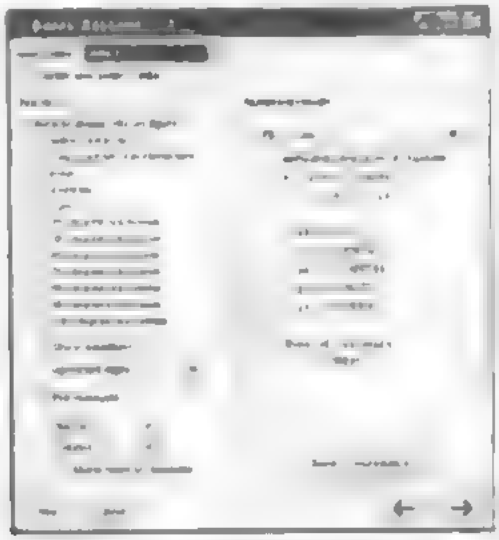


图 4.23　选择数据拟合的类型

STEP 7 [illegible]

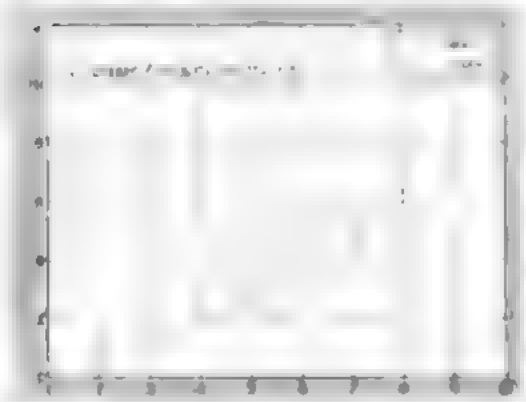


图 4.24　数据拟合的结果

如果选中此选项前的"Center and scale data"选项，MATLAB会将原始数据减去均值后的值（[illegible]），使拟合模型的性能[illegible]。[illegible]检测出系统使用了[illegible]，[illegible]。

4.3.2 绘制拟合残差图形

继续上面小节的步骤。

Step 1 绘制拟合残差图形。单击"Basic Fitting"窗口中的"[illegible]"选项，选择"Plot residuals"选项和"Show norm of residuals"选项，如图4.25所示。

图4.25 显示拟合残差以及其标准差

Step 2 查看绘制的结果。[illegible]，MATLAB会在原[illegible]下方绘制残差图形，并在图形中显示残差的标准差，绘制的结果如图4.26所示。

图4.26 显示拟合的残差

在"Basic Fitting"的选项中，可以选择残差图形的图表类型（Bar、Stem和Line），[illegible]；subplot和separate figure两个选项，在默认情况下，图表显示为bar，位置为subplot。

Step 3 保存拟合的结果。单击"Basic Fitting"对话框中的"Save to workspace"按钮，打开

"Save fit to workspace"（[illegible]），在其中设置保存选项，如图 4.27 所示。

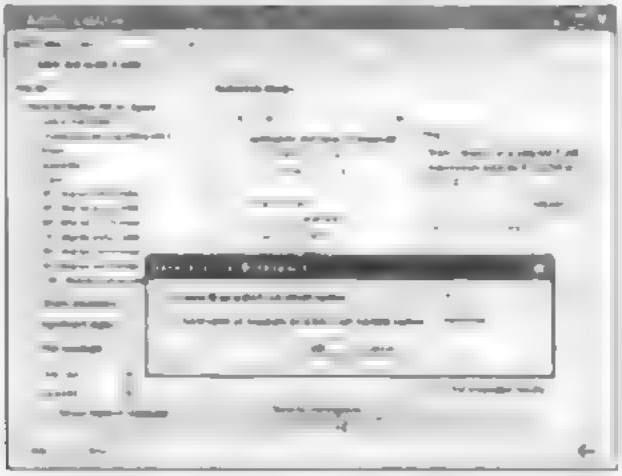


图 4.27　保存拟合的结果

Step 6 查看保存结果　"保存"[illegible]，[illegible]，查看保存结果。

```
Variables have been created in the current workspace.
>> fit1
fit1 =
     type: 'polynomial degree 3'
    coeff: [0.0098 -0.0607 0.7028 -0.5002]
>> normresid1
normresid1 =
    0.7882
```

4.3.3 进行数据预测

继续上面小节的步骤。

Step 1 创建数据　[illegible]，[illegible]，[illegible]，在"Enter values"[illegible]，[illegible]，[illegible]，[illegible]，最后，选择[illegible]，[illegible]，[illegible]显示在图形中，如图 4.28 所示。

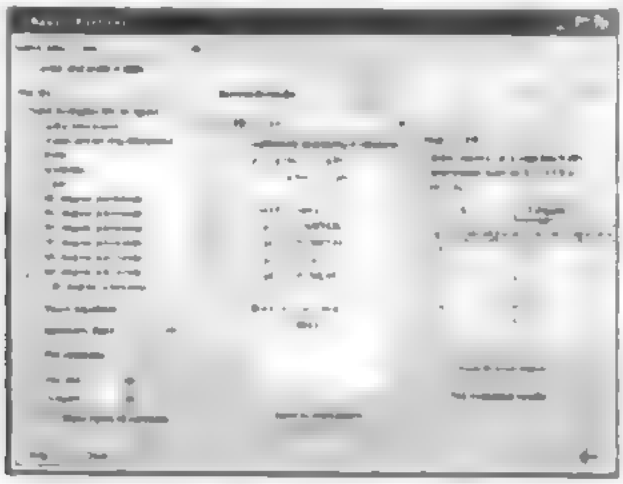


图 4.28　预测数据

Step 2 [illegible]

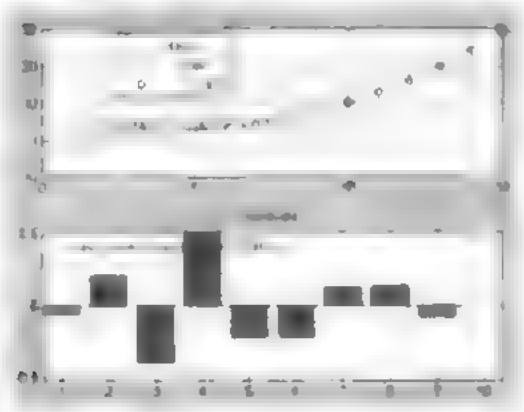


图 4.29 显示预测数据的图形

[illegible]

Step 3 [illegible]

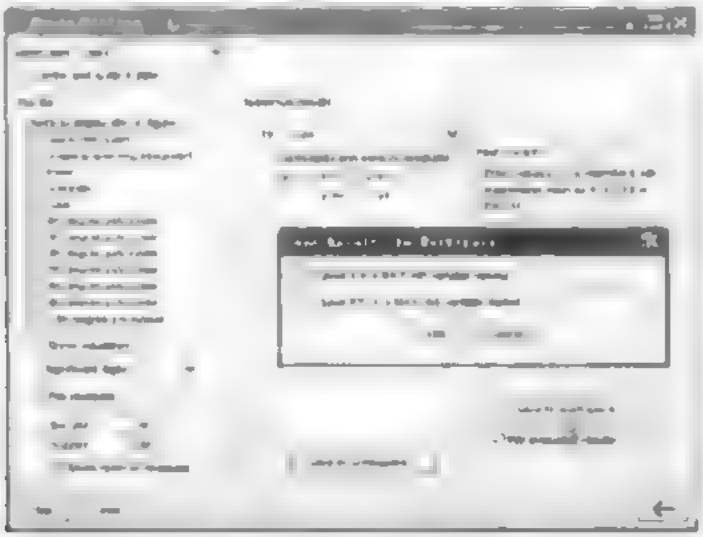


图 4.30 保存预测数据

Step 4 [illegible]

```
Variables have been created in the current workspace.
>> [ x1 fx]
ans =
   10.0000   10.2464
   11.0000   12.9151
   12.0000   16.1087
   13.0000   19.8459
```

```
14.0000    24.3055
15.0000    29.4262
```

[illegible]

4.4 傅里叶分析

[illegible]

4.4.1 离散 Fourier 变换

[illegible]

$$\text{DFT}: X(k)=\sum_{n=0}^{N-1}x[n]e^{-j2\pi kn/N} \qquad k=0,\ \ldots,N-1$$

$$\text{IDFT}: x[n]=\frac{1}{N}\sum_{k=0}^{N-1}X(k)e^{j2\pi kn/N} \qquad k=0,\ \ldots,N-1$$

[illegible]

- Y = fft(X,n,dim)
- y = ifft(X,n,dim)

[illegible]

[illegible]

例 4.12 使用FFT，从信号噪声中寻找隐藏在其中的频率成分并生成图表。

step 1 产生[illegible]，并绘制信号图形。在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> t = 0:0.001:0.6;
>>x = sin(2*pi*50*t)+sin(2*pi*120*t);
>>y = x + 2*randn(size(t));
>>plot(1000*t(1:50),y(1:50))
>>title('Signal Corrupted with Zero-Mean Random Noise')
```

```
>>xlabel('time (milliseconds)')
>> grid
```

Step 2 [illegible]

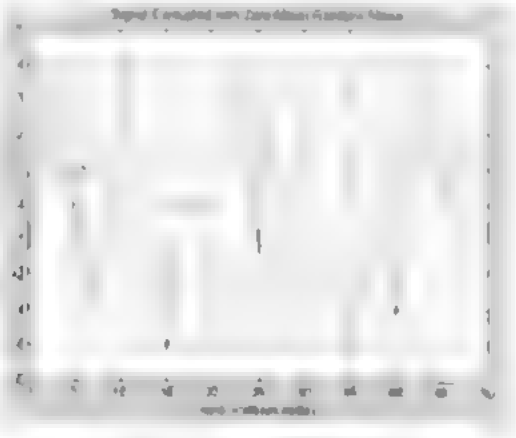


图 4.31 原始噪声信号

说明 [illegible]

Step 3 [illegible]

```
%进行Fourier变换
>> Y = fft(y,512);
>> Pyy = Y.* conj(Y) / 512;
>> f = 1000*(0:256)/512;
%打开新的图形窗口
>> figure
>> [illegible]
>> title('Frequency content of y')
>> [illegible]
>> [illegible]
```

Step 4 [illegible]

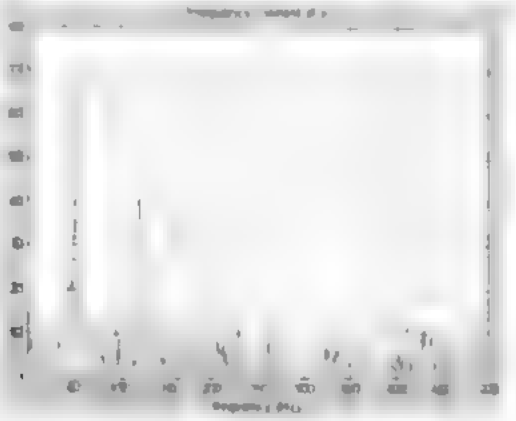


图 4.32 经过 Fourier 变换的信号

[illegible]

4.4.2 FFT 和 DFT

[illegible]

例 4.13 [illegible]

step 1 [illegible]

```
clear;
clf
[illegible]
[illegible]
x=[illegible]
tic
%使用DFT方法
for k=0:N-1
    X(k+1)=x*exp(-j*2*pi*k*n/N).';
end
    [illegible]
%使用IDFT方法
    [illegible]
        xz(n+1)=X*exp(j*2*pi*k*n/N).';
    end
        time_dft=toc;
        [illegible]
        [illegible]
        axis([0 1025 0 600])
        title('[illegible] Method')
        grid
        hold on
        tic
%使用FFT方法
        X1=fft(x);
%使用IFFT方法
        xr1=ifft(X1);
        time_fft=toc;
        subplot(212)
        plot(k,abs(X1),'r')
        axis([0 1025 0 600])
        title('FFT Method')
        grid
```

step 2 [illegible]

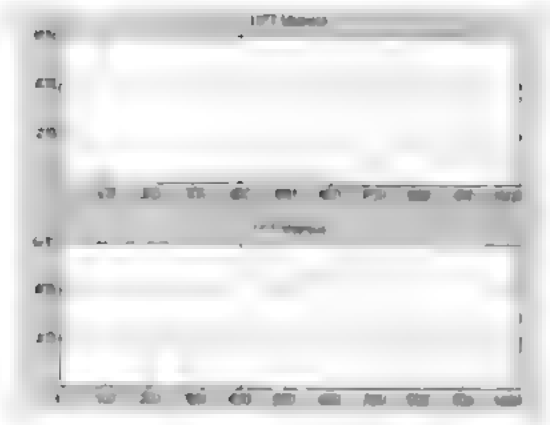


图 4.33 两种变换方法得到的结果

Step 3 [illegible]

```
>> s1 = ['The time of DFT Method is ' num2str(time_dft)];
>>s2 = ['The time of FFT Method is ' num2str(time_fft)];
[illegible]
```

Step 4 [illegible]

```
The time of DFT Method is 1.742
The time of FFT Method is 0.01
```

[illegible] 变换的时间是 0.01s。

[illegible]

4.4.3 DFT 的物理含义

[illegible] 号进行 DFT 转换。

其中连续信号的表达式如下：

$$x(t)=\sin(1.5\pi t)+0.5\cos(3\pi t)$$

[illegible] 最后绘制各种信号波形和转换图形。

例 4.14 [illegible] 最后绘制各种图形。

Step 1 [illegible]

```
[illegible]
[illegible]
w1=1.5*pi;
```

```
w2=3*pi;
N=32;
n=[0:N-1];
T=0.1;
t=n*T;
xan=sin(w1*t)+0.5*sin(w2*t);
subplot(421)
stem(t,xan,'.');
axis tight
grid
k=0:N-1;
Xa=fft(xan);
dscrp=norm(xan-real(ifft(Xa)));
subplot(423)
stem(k,abs(Xa),'.');
axis tight
grid

N=32;
n=[0:N-1];
T=0.05;
t=n*T;
xbn=sin(w1*t)+0.5*sin(w2*t);
subplot(422)
stem(t,xbn,'.');
axis tight
grid
k=0:N-1;
Xb=fft(xbn);
subplot(424)
stem(k,abs(Xb),'.');
axis tight
grid

N=64;
n=[0:N-1];
T=0.1;
t=n*T;
xcn=sin(w1*t)+0.5*sin(w2*t);
subplot(425)
stem(t,xcn,'.');
axis tight
grid
k=0:N-1;
Xc=fft(xcn);
subplot(427)
stem(k,abs(Xc),'.');
axis tight
grid

N=64;
n=[0:N-1];
T=0.05;
t=n*T;
xdn=sin(w1*t)+0.5*sin(w2*t);
subplot(426)
```

```
stem(t,xdn,'.');
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
stem(k,abs(Xd),'.');
axis tight
grid
```

step 2 [illegible]

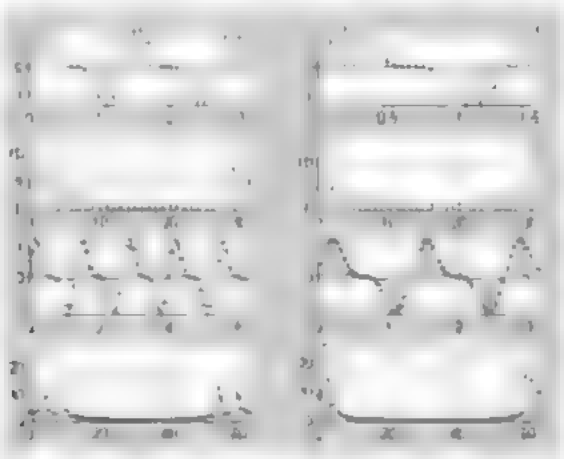


图 4.34 离散信号和 Fourier 转换后的图形

从上述信号处理来看，[illegible] 的影响，[illegible] 的处理。

4.4.4 使用 DFS 进行插值

MATLAB 提供一个专门的函数 [illegible]，[illegible] 数据增加采样，进行插值。

interpft 函数的主要调用格式如下：

- y = interpft(x,n)
- y = interpft(x,n,dim)

[illegible] 中，dim 则表示这种插值要在指定的维度上进行。

[illegible]

[illegible]，[illegible] 数据插值，[illegible] 数据插值，[illegible]

从信号序列$X(k)$中选择数据点，并使用下面的公式进行插值运算：

$$\tilde{x}(t)=\frac{1}{N}\sum_{|k|<N/2}\tilde{X}(k)e^{j2\pi kt/NT}$$

$$=\frac{1}{N}\{X(0)+2\sum_{k=1}^{N/2-1}Real(X(k)e^{j2\pi kt/NT}]+X(N/2)\cos(\pi t/T)\}$$

根据上面的公式，读者可以自行编写M文件，然后使用编写的文件进行插值运算。

例 4.15 根据上面的基础插值公式，自行编写使用DFS方法进行插值的M文件，然后以$f(t)=\sin(\pi t)+0.5\sin(0.5\pi t)$为主信号，以$z(t)=rand(1,n)-0.5$为噪声信号为数据信号源，产生基础数据，最后使用DFS进行插值。

step 1 选择命令窗口编辑栏中的“File”⇨“New”⇨“M-File”命令，打开M文件编辑器，在其中输入下面的程序代码：

```
function [xi,yi]=interp_DFS(T,x,Ws,ti)
%T: 样本间隔
%x: 离散时间样本
%Ws:归一化的频率
%ti: 插值的时间系列

if nargin<4
    ti=5;
end
if nargin<3 | Ws>1
    Ws=1;
end
N=length(x);
if length(ti)==1
    ti=0:T/ti:(N-1)*T; %使用ti得到的细化数据系列
end
ks=ceil(Ws*N/2);
yi=fft(x);
yi(ks+2:N-ks)=zeros(1,N-2*ks-1);    %筛选后的时间系列
xi=zeros(1,length(ti));
for k=2:N/2
    xi=xi+yi(k)*exp(j*2*pi*(k-1)*ti/N/T);
end
xi=real(2*xi+yi(1)+yi(N/2+1)*cos(pi*ti/T))/N;   %根据公式计算插值结果
```

在输入上面的程序代码后，将上面的程序代码保存为“interp_DFS.m”文件，在后面的步骤中，将会使用该函数进行数据插值。

step 1 在MATLAB的命令窗口中输入下面的程序代码：

```
T=0.1;
N=32;
ti=[0:T/5:(N-1)*T];
w1=pi;
w2=0.5*pi;
n=[0:N-1];
t=n*T;
x=sin(w1*t)+0.5*sin(w2*t)+rand(1,N)-0.5;
%绘制原始数据图形
```

```
[illegible]
[illegible]
title('Original and interpolation signal')
grid on;
axis tight
hold on;
%进行DFS数据插值
[illegible]
[illegible]
k=[0:N-1];
%绘制spectrum图形
subplot(412)
[illegible]
[illegible]
[illegible]
box on
%改变参数，重新进行DFS数据插值
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
box on
%绘制插值后波形
subplot(414)
plot(t,x,'k.',ti,xi,'r')
[illegible]
grid
axis tight
```

step 3 [illegible]

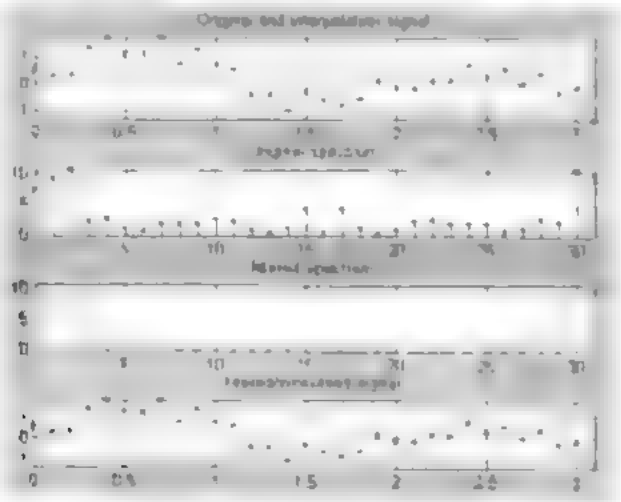


图 4.35 对 DFS 进行数据插值

[illegible]

4.5 优化

[illegible]

$$\min_{x \in S} f(x)$$

[illegible]

[illegible] Toolbox 软件。

4.5.1 无约束非线性优化命令

[illegible] fminsearch 和 fminunc 两种函数，其对应的详细调用格式如下

```
[x,fval,exitflag,output] = fminsearch(fun,x0,options)
```

在上面的命令格式中，参数比较繁多，下面分别分详细介绍。

- **输入参数**：参数 [illegible]
- **输出参数**：参数 [illegible]

[illegible]

fminunc 函数的调用格式如下：

```
[x,fval,exitflag,output,grad,hessian] = fminunc(fun,x0,options)
```

- [illegible]

4.5.2 无约束非线性优化实例

[illegible]

例 4.16 求解二元函数 $f(x)=3x_1^2+2x_1x_2+x_2^2$ 在全集范围之内的最小值，分别使用不同的优化函数和优化属性。为了让读者能够直观地查看优化求解情况，可以在求解之后绘制一元函数的图形。

step 1 选择命令窗口编辑栏中的"File" ⇨ "New" ⇨ "M-File"命令，打开M文件编辑器，在其中输入下面的程序代码：

```
function [ f,g] = optfun(x)
f = 3*x(1)^2 + 2*x(1)*x(2) + x(2)^2;
if nargout > 1
   g(1) = 6*x(1)+2*x(2);
   g(2) = 2*x(1)+2*x(2);
end
```

在上面的程序代码中，g表示的是f函数的偏导数，满足下面的方程组：

$$\begin{cases} g(1)=\dfrac{\partial f(x)}{\partial x_1}=6x_1+2x_2 \\ g(2)=\dfrac{\partial f(x)}{\partial x_2}=2x_1+2x_2 \end{cases}$$

在输入完上面的程序代码后，将该代码保存为"optfun.m"文件。

step 2 选择优化的初始数值[1,1]，分别使用不同的函数求解优化。在MATLAB的命令窗口中输入下面的程序代码：

```
>> x0=[ 1,1];
>> options = optimset('Display','iter','TolFun',1e-18,'GradObj','on');
>> [ x,fval,exitflag,output,grad] =fminunc(@optfun,x0,options)
                                  Norm of      First-order
 Iteration        f(x)            step         optimality   CG-iterations
     0                   6                                8
     1        6.28624e-031        1.41421      1.55e-015            1
     2        2.91704e-062    8.9509e-016      5.92e-031            1
Optimization terminated: first-order optimality less than OPTIONS.TolFun,
 and no negative/zero curvature detected in trust region model.
x =
  1.0e-031 *
   -0.9861         0
fval =
  2.9170e-062
exitflag =
     1
output =
       iterations: 2
        funcCount: 3
     cgiterations: 2
    firstorderopt: 5.9165e-031
        algorithm: 'large-scale: trust-region Newton'
          message: [ 1x137 char]
grad =
  1.0e-030 *
   -0.5916
   -0.1972
>> [ x1,fval1,exitflag1,output1] = fminsearch(@optfun,x0,options)
  Iteration   Func-count     min f(x)         Procedure
```

```
     [illegible]          1                 6
     [illegible]          3                 6          initial simplex
                          5           5.52062          expand
     [illegible]          7           4.91391          expand
     [illegible]          9           3.86566          expand
                         11           2.52517          expand
    ..................  ..........................//限于篇幅,省略了部分数据
     [illegible]        151      1.95216e-018         contract inside
     [illegible]        153      4.03648e-019         contract inside
     [illegible]        155      4.03648e-019         contract inside
     [illegible]        157      4.03648e-019         contract inside
x1 =
 1.0e-009 *
   -0.4052    0.1308
fval1 =
  [illegible]
exitflag1 =
     [illegible]
output1 =
     [illegible]: 81
     funcCount: 157
     [illegible]: 'Maximum number of function evaluations exceeded'
       message: [1x196 char]
```

[illegible]

Step 3 [illegible] 下面的程序代码

```
>> x0=[-1,1];
>> options = optimset('Display','iter','TolFun',1e-18,'GradObj','on');
>> [x,fval,exitflag,output,grad] =fminunc(@optfun,x0,options)
                                    Norm of      First-order
 [illegible]                          [illegible]                  [illegible]
     0                 2                                   4
     1          0.666667        0.666667            1.33            1
     2          0.222222        0.666667            1.33            1
     3         0.0740741        0.222222           0.444            1
     4         0.0246914        0.222222           0.444            1
     5        0.00823045       0.0740741           0.148            1
    ..........................................................//限于篇幅,省略了部分数据
    25       2.36047e-012    1.25445e-006       2.51e-006           1
    26       7.86824e-013    1.25445e-006       2.51e-006           1
    27       2.62275e-013     4.1815e-007       8.36e-007           1
 [illegible]
x =
  1.0e-006 *
   -0.2091    0.6272
fval =
  2.6227e-013
```

```
exitflag =
     1
output =
        iterations: 27
         funcCount: 28
     cgiterations: 27
    firstorderopt: 8.3630e-007
        algorithm: 'large-scale: trust-region Newton'
          message: [illegible]
grad =
  1.0e-006 *
    0.0070
    0.8360
>> [x1,fval1,exitflag1,output1] = fminsearch(@optfun,x0,options)
  Iteration   Func-count     min f(x)         Procedure
     0            1                2
     1            3                2         initial simplex
     2            5          1.65062         expand
     3            7          1.47141         expand
     4            9          [illegible]     [illegible]
     5           11         0.766885         expand
     6           13         0.766885         contract outside
     7           15         0.766885         contract outside
     8           17         0.602196         expand
     9           18         0.602196         reflect
10           20         0.334623         expand

    [illegible]                              //限于篇幅,省略了部分数据
    74          143     3.83172e-019         contract outside
    75          145     2.24524e-019         contract inside
    76          147     2.24524e-019         contract inside
x1 =
  1.0e-009 *
   -0.3318    0.3981
fval1 =
  [illegible]
exitflag1 =
     1
output1 =
    iterations: 76
     funcCount: 147
     algorithm: 'Nelder-Mead simplex direct search'
       message: [1x196 char]
```

从上面的计算结果可以看出，fminsearch函数在给定初始条件下，会根据目标函数的变化情况自动调整搜索方向，最终收敛到函数的最小值点附近。

在使用fminsearch函数求解多维优化问题时，需要使用optimset函数来设置搜索参数，以提高搜索效率和精度。请自行查阅帮助文档。

Step 4 在[-1.5, 1.5]区间内绘制二维函数 $f(x)=3x_1^2+2x_1x_2+x_2^2$ 的曲线，并以步长0.01计算函数数值。在MATLAB的命令窗口中输入下面的程序代码

```
>> x=-1.5:0.02:1.5;
>>y=x;
[illegible]
[illegible]
[illegible]
>>shading interp
>> colorbar horiz
```

Step 3 [illegible]

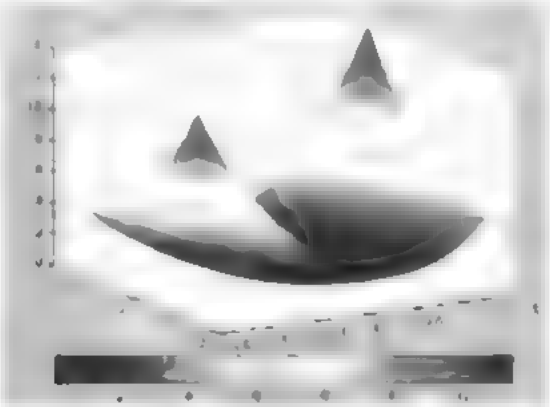


图 4-36 三维函数图形

[illegible]

4.5.3 非线性最小方差命令

非线性最小方差（nonlinear least squares）是一种非线性优化的一种特殊情况，其优化目标函数 $\mathrm{Min}\sum_{i=1}^{n} f_i(x)$ [illegible] 这种数学模型称为非线性数学模型（nonlinear data-fitting）。

在 MATLAB 中，[illegible] 提供 lsqnonlin 函数命令，其最完整的调用格式如下：

[illegible]

命令在命令中，函数参数比较复杂，下面主要介绍参数的基本含义。

- **输入参数**：参数 fun 表示优化的目标函数，参数 x0 表示优化求解的初值条件，也是进行优化求解的上限，也是进行优化求解的上限，性质等于 lb≤x≤ub。参数 options 则表示进行优化求解的优化属性。
- **输出参数**：参数 x 表示以求解的最优解，参数 resnorm 则表示残差平方和，在数值上等于 sum(residual^2)。参数 residual 则表示优化求解后的残差，参数 lambda 则表示函数在最优解处的拉格朗日数值。其他参数与其他优化命令中的含义相同。

[illegible]

4.5.4 非线性最小方差实例

例 4.17 以函数 $f(x)=\frac{1}{1+8x^2}$ [illegible] 为目标进行数据拟合。

Step 1 [illegible] 中输入下面的程序代码

```
function F=flq(a)
xx=-2+[0:200]/50;
F=polyval(a,xx)-1./(1+8*xx.*xx);
```

[illegible]

Step 2 [illegible]

```
[illegible]
[illegible]
lb=-1*ones(1,N);
[illegible]
[illegible]
```

Step 3 [illegible] 键，[illegible]

```
                                        Norm of      First-order
 Iteration  Func-count     f(x)          step         optimality   CG-iterations
     0          6       27.7062                          49.4
     1         12       12.2685       0.261626           9.63            2
     2         18       9.81249        0.16763           18.1            2
     3         24       6.76791       0.195781           4.87            2
     4         30       5.94883       0.105258           7.67            2
     5         36       4.93127       0.149942           3.48            2
     [illegible]
    32        198       3.89864   3.62219e-006        6.6e-005           2
    33        204       3.89864   2.94793e-006       8.93e-005           2
    34        210       3.89864   1.63203e-006       6.33e-005           2
    35        216       3.89864   2.08912e-006       7.21e-005           2
    36        222       3.89864   9.19467e-007       2.77e-005           2
>> x
x =
    0.1124    0.0000   -0.5522    0.0000    0.6225
>> resnorm
resnorm =
    3.8986
>> exitflag
exitflag =
```

```
      2
>> output
output =
    firstorderopt: 2.[illegible]e-05
       iterations: 36
        funcCount: 222
     cgiterations: 71
          algorithm: 'trust-region-reflective Newton'
          message: [1x77 char]
>> lambda.lower
ans =
    [illegible]
    [illegible]
    [illegible]
>> lambda.upper
ans =
    0
    0
    0
    0
    0
```

step 4 计算非线性最小二乘拟合的数据残差，在命令窗口中输入下面的语句：

```
>> xx=-2+[0:200]/50;
>> plot(xx,residual,'r','LineWidth',1.5)
>> grid
>> title('The residual of data')
```

step 5 查看残差，在输入语句后按"Enter"键，得到如图4.37所示的图形。

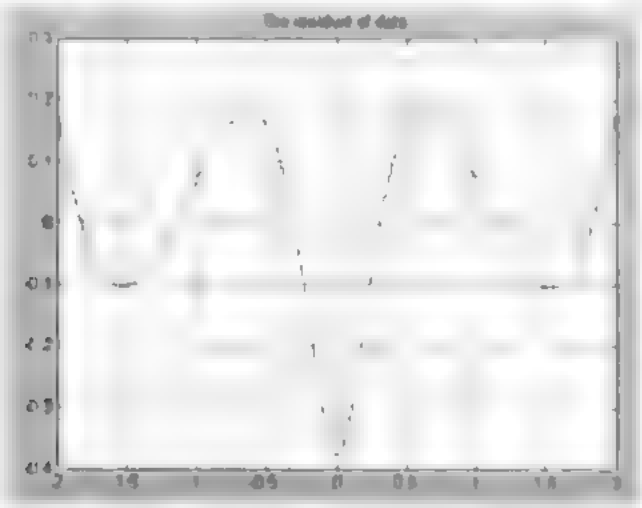


图4.37 数据残余的图形

说明

[illegible]

455 约束条件的非线性优化命令

[illegible]

$$\text{Min} \quad f(x)$$

$$\text{Subject to} \quad c(x) \le 0$$

$$ceq(x) = 0$$

$$A \cdot x \le b$$

$$Aeq \cdot x = beq$$

$$lb \le x \le ub$$

该函数的完整调用格式如下：

[illegible]

该函数的参数比较复杂，下面详细介绍各种参数的含义。

- **输入参数**：[illegible] Aeq、beq 分别[illegible]；参数 lb、ub 分别[illegible] $lb \le x \le ub$ 中[illegible]；参数 [illegible] $c(x) \le 0$ 和 $ceq(x)=0$ 的[illegible]；参数 options 和 nonlcon 都是进行优化的[illegible]。
- **输出参数**：[illegible]

[illegible]

456 约束条件的非线性优化实例

例4.18 [illegible] $0 \le x_1 + 2x_2 + 3x_3 \le 72$，[illegible] $f(x) = -x_1x_2x_3$ 的最小值（即最优解的数值）。

STEP 1 [illegible]

$$-x_1 - 2x_2 - 3x_3 \le 0$$

$$x_1 + 2x_2 + 3x_3 \le 72$$

[illegible]

step 2 选择命令窗口编辑栏中的"File" ⇨ "New" ⇨ "M-File"命令，打开M文件编辑器，在其中输入下面的程序代码：

```
function f=optcon(x)
f=-x(1)*x(2)*x(3);
```

将上面的程序代码保存为"optcon.m"文件，该文件将是最优解的目标函数。

step 3 在MATLAB的命令窗口中输入下面的程序代码：

```
>> A=[1,-2,-2;1,2,2];
>> b=[0;72];
>> x0=[10;10;10];
>>[x,fval,exitflag,output,lambda] = fmincon(@optcon,x0,A,b);
```

step 4 查看优化信息。在输入上面的程序代码后，按"Enter"键，MATLAB将会进行优化运算，并显示对应的优化信息：

```
Optimization terminated: Magnitude of directional derivative in search
 direction less than 2*cptions.TolFun and maximum constraint violation
  is less than options.TolCon.
Active inequalities (to within options.TolCon = 1e-006):
  lower      upper     ineqlin   ineqnonlin
                          2
```

在上面的程序中显示了实质起作用的约束条件和优化中上的类型。

step 5 查看优化的结果。在MATLAB的命令窗口中输入下面的程序代码：

```
>> x
x =
   24.0000
   12.0000
   12.0000
>> fval
fval =
 -3.4560e+003
>> exitflag
exitflag =
     5
>> output
output =
       iterations: 8
        funcCount: 48
         stepsize: 1
        algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
    firstorderopt: 1.5459e-004
     cgiterations: []
          message: [1x172 char]
>> lambda
lambda =
         lower: [3x1 double]
         upper: [3x1 double]
         eqlin: [0x1 double]
      eqnonlin: [0x1 double]
```

```
        ineqlin: [ 2x1 double]
     ineqnonlin: [ 0x1 double]
>> lambda.lower
ans =
     [illegible]
     [illegible]
     [illegible]
>> lambda.upper
ans =
     [illegible]
     [illegible]
     [illegible]
```

从[illegible]，可以看[illegible]，这[illegible]。由于在[illegible]的[illegible]，没有[illegible]，在[illegible]参数中[illegible]

Step 6 单击[illegible]，进行[illegible]。此时[illegible]

$$x_1 \geq 5$$
$$x_1 \leq 10$$
$$x_1 + 2x_2 + 3x_3 \leq 72$$

[illegible]，将初值设置为[1, 1, 1]，[illegible]，程序[illegible]

```
>> x0=[1,1,1];
>> A=[1,2,2;0,1,0;0,0,1];
 b=[72;5;10];
>> [x,fval,exitflag,output,lambda,grad,hessian]=fmincon(fun,x0,A,b);
Warning: [illegible] (line [illegible]) [illegible] not [illegible]
this type of problem,
 switching to medium-scale (line search).
> In fmincon at 260
[illegible]
 than options.TolFun and maximum constraint violation is less
 than options.TolCon.
Active inequalities (to within options.TolCon = 1e-006):
  lower       upper      ineqlin   ineqnonlin
                             1
                             2
                             3
>> x
x =
   42.0000    5.0000   10.0000
>> fval
fval =
   [illegible]
>> exitflag
exitflag =
     1
>> output
output =
        iterations: 5
         funcCount: 29
         [illegible]
```

```
        [illegible]
    firstorderopt: 0
        [illegible]
          message: [1x144 char]
>> lambda
lambda =
          lower: [3x1 double]
          upper: [3x1 double]
          eqlin: [0x1 double]
        eqnonlin: [0x1 double]
         ineqlin: [3x1 double]
      ineqnonlin: [0x1 double]
>> lambda.lower
ans =
     [illegible]
     [illegible]
     [illegible]
>> [illegible]
>> [illegible]
     [illegible]
```

从上面的结果可以看出，当使用不同的优化算法时，得到的解会有一定的差异，因此在实际应用中多解决问题时，需要将不同的算法结果进行比较。

4.5.7 最小最大值的优化问题

[illegible] 值，相当于求解下面的优化问题

$$\min_{x} \max_{(F)} \{F_i(x)\}$$

$$\text{Subject to} \quad c(x) \le 0$$

$$ceq(x) = 0$$

$$A \cdot x \le b$$

$$Aeq \cdot x = beq$$

$$lb \le x \le ub$$

[illegible]

例 4.19 [illegible] 函数 $F(x) = [f_1(x), f_2(x), f_3(x), f_4(x), f_5(x)]^T$ 的最小最大值，其中多目标函数分别为 $f_1(x) = 2x_1^2 + x_2^2 - 48x_1 - 40x_2 + 125$，$f_2(x) = x_1^2 - 3x_2$，$f_3(x) = x_1 + 3x_2 - 18$，$f_4(x) = -x_1 - x_2$，$f_5(x) = x_1 + x_2 - 8$。

Step 1 [illegible] 中输入下面的程序代码

```
function f = mnmax(x)
[illegible]
```

```
f(2)= -x(1)^2 - 3*x(2)^2;
f(3)= x(1) + 3*x(2) -18;
f(4)= -x(1)- x(2);
f(5)= x(1) + x(2) - 8;
```

在输入上面的程序代码后，将该代码保存为“mnmax.m”文件。

step 2 求解最小最大值的优化问题。在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> x0 = [ 0.1; 0.1];
>> [ x,fval] = fminimax(@mnmax,x0)
Optimization terminated: Search direction less than 2*options.TolX
 and maximum constraint violation is less than options.TolCon.
Active inequalities (to within options.TolCon = 1e-006):
  lower      upper     ineqlin   ineqnonlin
                                      4
                                      5
x =
    1.3333
    2.6667
fval =
  -34.9994  -23.1121   -8.6665   -4.0000   -4.0000
```

在上面的求解过程中，首先设置了初值条件，然后直接调用函数求解优化问题。

step 2 重新设置优化条件，求解优化问题。在命令窗口中输入下面的程序代码：

```
>> x0 = [ 0.1; 0.1];        % Make a starting guess at solution
options = optimset('MinAbsMax',5);   % Minimize absolute values
[ x,fval,maxfval,exitflag,output,lambda] = fminimax(@mnmax,x0,[],[],[],
[],[],[],[],options)
Optimization terminated: Search direction less than 2*options.TolX
 and maximum constraint violation is less than options.TolCon.
Active inequalities (to within options.TolCon = 1e-006):
  lower      upper     ineqlin   ineqnonlin
                                      6
                                      7
                                      8
x =
    1.5768
    1.7126
fval =
  -11.2854  -11.2854  -11.2854   -3.2894   -4.7106
maxfval =
   11.2854
exitflag =
     4
output =
       iterations: 9
        funcCount: 49
         stepsize: 1
        algorithm: 'minimax SQP, Quasi-Newton, line_search'
    firstorderopt: []
     cgiterations: []
          message: [ 1x129 char]
lambda =
```

```
          lower: [2x1 double]
          upper: [2x1 double]
          eqlin: [0x1 double]
       eqnonlin: [0x1 double]
        ineqlin: [0x1 double]
     ineqnonlin: [0x1 double]
```

[illegible]

Step 5 命令行窗口，[illegible]输入 "optimset fminimax"，查看该函数的所有相关优化属性如下

```
>> optimset fminimax
ans =
                  Display: [illegible]
              MaxFunEvals: '100*numberofvariables'
                  MaxIter: 400
                   TolFun: 1.0000e-006
                     TolX: 1.0000e-006
              FunValCheck: [illegible]
                [illegible]
          ActiveConstrTol:
       NoStopIfFlatInfeas:
           BranchStrategy:
                [illegible]: [illegible]
              Diagnostics: [illegible]
                [illegible]: 0.1000
                [illegible]: 1.0000e-009
                [illegible]: []
              GradConstr: [illegible]
                [illegible]: [illegible]
                 Hessian: [illegible]
                [illegible]: ..//限于篇幅,这里省略了部分属性列表
                [illegible]: []
            RelLineSrchBnd: []
     RelLineSrchBndDuration:
                [illegible]: []
                [illegible]:
                  TolCon: [illegible]
                  TolPCG: []
               TolPLPFun: []
                [illegible]: []
               TypicalX: 'ones(numberofvariables,1)'
```

[illegible]

4.5.8 对比实例

例 4.20 使用函数 $f(x)=\frac{1}{1+8x}$ 产生的样本数据，[illegible]

step 1 在 MATLAB 的命令窗口，输入如下命令：

```
clear
clc
%创建各个内取函数
[illegible]
[illegible]
[illegible]
%数据拟合的阶数
N=2;
[illegible]
%产生拟合的数据系列
xx=-2:(0:200)'/50;
[illegible]
%使用最小最大值的方法进行拟合
[illegible]
%使用最小方差的方法进行拟合
[illegible]
%绘制拟合的数据曲线
[illegible]
hold on;
plot(xx,polyval(fm,xx),'m','LineWidth',1.5);
hold on;
plot(xx,polyval(fln,xx),'g','LineWidth',1.5)
[illegible]
axis([-2 2 -0.4 1.1])
[illegible]
title('The Curve Fitting')
```

step 2 查看[illegible]，输入[illegible]，按 Enter 键，[illegible]

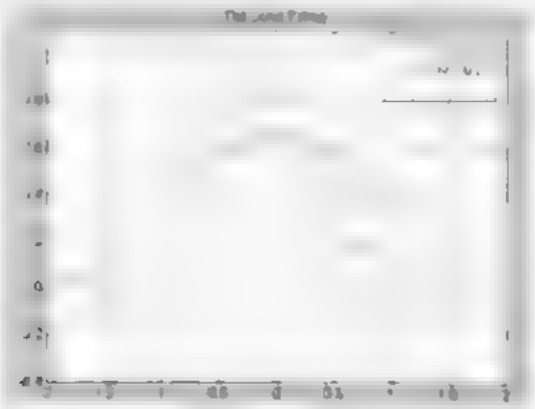


图 4.38 绘制数据拟合的结果

step 3 查看[illegible]

[illegible]

```
 and maximum constraint violation is less than options.TolCon.
Active inequalities (to within options.TolCon = 1e-006):
  lower      upper     ineqlin   ineqnonlin

                                     62
                                    101
                                    140
                                    261
[illegible]
 and no negative/zero curvature detected in trust region model.
>> fm
fm =
   -0.2424    0.0000    0.6530
[illegible]
fln =
   -0.1631   -0.0000    0.4653
```

[illegible]

4.5.9 线性规划

[illegible]使用比较特殊的方法来求解。典型的线性规划问题为

$$\min \quad f^T x$$

$$\text{Subject to} \quad A \cdot x \le b$$

$$Aeq \cdot x = beq$$

$$lb \le x \le ub$$

在MATLAB中，求解线性规划的[illegible]

◆ [illegible] = linprog(f,A,b,Aeq,beq,lb,ub,[illegible])

关于该命令中的各参数的含义，请参考前面的fmincon命令。

例4.21 [illegible]，其中目标函数为 $f(x)=3x_1-2x_2$，[illegible]表示为 $0\le x_1,x_2\le 10$。同时，该目标函数满足下面的约束条件

$$2x_1+x_2\le 3$$
$$3x_1+4x_2\le 7$$
$$-3x_1+2x_2\le 2$$

[illegible]上面的规划问题。

Step 1 [illegible]

```
>> x0=[0 0]';
[illegible]
```

```
[illegible]
[illegible];
Aeq=[-3 2];
[illegible];
l=[0 0];
u=[10 10];
tic
%使用线性规划的方法来求解优化问题
[x,fval,exitflag,output,lambda] =linprog(f,A,b,Aeq,beq,l,u);
time_lin=toc;
cons_st=[A;Aeq]*x-[b;beq];
[illegible]
[illegible]
[illegible]
```

step 2 在命令窗口中输入[illegible]，分别查看[illegible]

```
>> x,fval
x =
    0.3333
    1.5000
fval =
   -4.0000
>> lambda.ineqlin
ans =
    0.6667
    0.0000
>> lambda.eqlin
ans =
   -0.4333
>> lambda.upper
ans =
  1.0e-011 *
    0.2480
    [illegible]
>> lambda.lower
ans =
  1.0e-010 *
    0.8574
0.0700
>> cons_st
cons_st =
    [illegible]
    [illegible]
    [illegible]
```

说明 小[illegible]中可以看出，[illegible]，[illegible]，[illegible]

step 3 在命令窗口中输入[illegible]，[illegible]

```
>> xc,fvalc
xc =
    0.3333    1.5000
```

```
fvalc =
    -4
>> lambdac.ineqlin
ans =
    0.6667
>> lambdac.eqlin
ans =
   -0.3333
```

从上面步骤的结果中可见，通过使用约束条件求解得到的结果和线性规划的结果一致。同时，关于各种约束条件的拉格朗日乘数也可以一目了然。

Step 4 对比两种方法的使用时间。在命令窗口中输入下面的程序代码

```
>> s1 = ['The time of Linear Method is ' num2str(time_lin)];
>> s2 = ['The time of Constrained Method is ' num2str(time_con)];
>>S = strvcat(s1,s2);
>>disp(S)
The time of Linear Method is 0.09
The time of Constrained Method is 0.13
```

从上面的结果中可以看到，使用线性规划求解的时间，远远小于使用约束条件求解的时间。但是线性规划要求目标、约束条件全部为线性的才可以。

在求解一个优化问题时，针对不同的问题选择合适的方法，通过这种方法，可以使问题的结构变得简单明了。如图4.39所示。

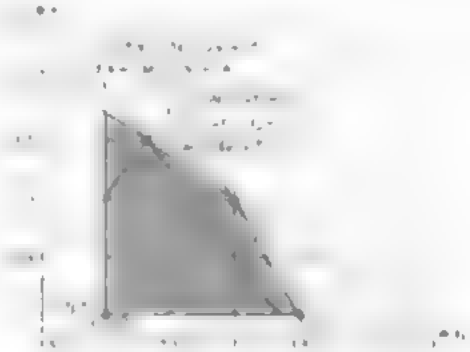


图4.39 线性规划的几何含义

4.5.10 二次规划

二次规划是指其目标函数最高为两个变量的优化问题，典型的二次规划的模式如下

$$\min \frac{1}{2}x^T Hx + f^T x$$

$$\text{Subject to} \quad A \cdot x \le b$$

$$Aeq \cdot x = beq$$

$$lb \le x \le ub$$

在 MATLAB 中，求解线性规划的命令为 quadprog，其完整的调用格式如下：

[x,fval,exitflag,output,lambda] = quadprog (f,A,b,Aeq,beq,lb,ub,x0,options)

关于该命令中的各参数的含义，请参考前面的 fmincon 命令。

例 4.22 求解二次规划，其目标函数为 $f(x)=\frac{1}{2}x_1^2+x_2^2-x_1x_2-2x_1-6x_2$，同时，其满足的约束条件为：

$$\begin{cases} x_1+x_2\leqslant 2 \\ -x_1+2x_2\leqslant 2 \\ 2x_1+x_2\leqslant 3 \\ 0\leqslant x_1,x_2 \end{cases}$$

step 1 将二次规划进行转换，转换为标准形式。根据线性代数知识，得到的结果为：

$$H=\begin{pmatrix}1 & 1\\ -1 & 2\end{pmatrix},\quad f=\begin{pmatrix}-2\\ -6\end{pmatrix},\quad x=\begin{pmatrix}x_1\\ x_2\end{pmatrix}$$

step 2 进行二次规划求解。在 MATLAB 的命令窗口中输入下面的代码：

```
%转换成标准形式
>>H = [ 1 -1; -1 2] ;
>>f = [ -2; -6] ;
>>A = [ 1 1; -1 2; 2 1] ;
>>b = [ 2; 2; 3] ;
>>lb = zeros(2,1);
>>[ x,fval,exitflag,output,lambda] = quadprog(H,f,A,b,[ ],[ ],lb);
```

step 3 查看二次规划求解的结果。在 MATLAB 的命令窗口中输入代码：

```
>> x
x =
    0.6667
    1.3333
>> fval
fval =
   -8.2222
>> lambda.ineqlin
ans =
    3.1111
    0.4444
         0
```

在上面的二次规划问题中，求得的最优解为 (0.6667，1.3333)，对应的函数数值为 −8.222。同时，对应的拉格朗日系数为 (3.1111，0.4444，0)。

使用遗传算法求解二次规划

在本节的最后，将介绍如何使用遗传算法来求解这个二次规划问题。如果希望使用本节中的方法，请首先安装 "Genetic Algorithm and Direct Search Toolbox"，遗传算法是近年来发展迅速的计算方法，将在后面章节中详细介绍。下面分步骤介绍如何使用该方法来求解上面的实例。

例 4.23　使用遗传算法来求解上面的二次规划。

step 1 选择主菜单"File"→"New"→"M-file"命令，打开M文件编辑器，在其中输入下面的程序代码

```
function y = quadopt(x);
x = x';
H = [ 1 -1; -1 2];
f = [-2, -6];
y = 0.5*x'*H*x + f'*x;
```

[illegible]

step 2 设置[illegible]，在MATLAB命令窗口中输入下面的代码：

```
>> A = [ 1 1; -1 2; 2 1];
>> b = [ 2; 2; 3];
>> lb = zeros(2,1);
>> x0=[ 0,0];
```

step 3 设置优化参数。在命令窗口中输入"optimtool"，打开"Pattern Search Tool"对话框，并在其中设置规划的参数，如图 4.40 所示。

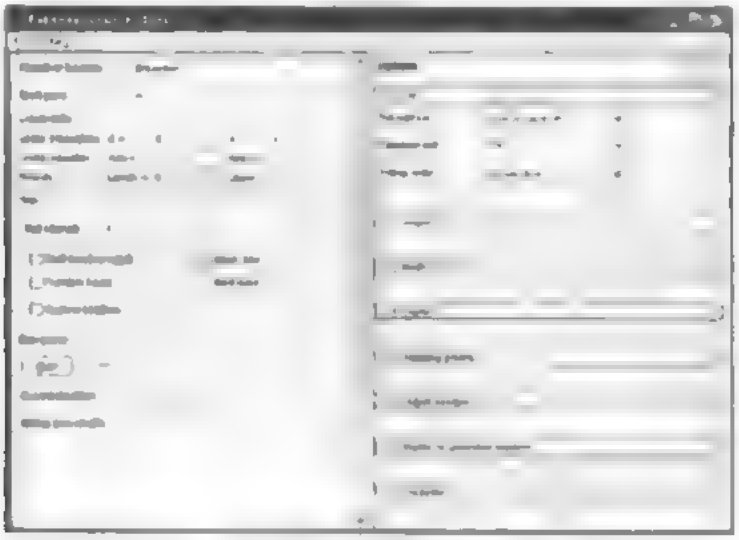


图 4.40　设置优化条件

在上面的对话框中，分别设置了二次规划的参数

- 在"Objective function"选项中输入"@quadopt"，其中quadopt就是在前面步骤中设置的二次规划的目标函数
- 在"Start point"选项中输入"x0"，表示进行优化计算时的初始条件数值，在前面使用quadopt函数时，并没有设置这个选项，但在使用遗传算法时，必须指定初始条件的数值
- 在"Constraints"栏目中设置[illegible]约束条件，其中"Linear inequalities"选项表示不等式的约束条件，可以在"A"和"b"选项中输入相应的参数。"Linear equalities"选项用来表示等式的约束条件，在本实例中没有任何等式约束，因此将这个选项中的"Bounds"选项表示变量的取值范围$lb \le x \le ub$，在本实例中，只有下限，没有上限，因此在"Lower"选项中输入"lb"。

Step 3 [illegible]，并进行求解，得出对应的结果，如图 4.41 所示。

图 4.41 规划求解的结果

[illegible]求解结果，具体信息如下

```
[illegible]
Pattern Search terminated.
Objective function value: -0.22222052697175
Optimization terminated: current mesh size 9.5367e-007 is less than
[illegible]
```

[illegible]了优化求解过程的步骤

[illegible]使用 quadprog 方法求解得到的最优解相同。

4.6 使用遗传算法求解优化

[illegible]algorithm 方法，基于自然选择的启发式搜索算法，[illegible]遗传算法。在遗传算法中，将要解决的问题[illegible]，在每个[illegible]，[illegible]，使用[illegible]。在多变量连续的数据分析中，将[illegible]

在 MATLAB 中，[illegible]遗传算法求解[illegible]，[illegible]，使用遗传算法求解[illegible]

前面介绍的优化方法更有效、更方便。

[illegible] 在本节中，将[illegible]采用"Genetic Algorithm and Direct Search"工具箱中提供的函数来进行优化求解。

在"Genetic Algorithm and Direct Search"工具箱中提供了[illegible]，方便用户使用[illegible]。如果要查看"解决[illegible]"，可以在命令窗口中输入"help optimdemo"来查看其[illegible]。

4.6.1 分析目标函数

例 4.24 使用遗传算法来求解优化问题。本例分别采用"Genetic Algorithm and Direct Search"工具箱中的适应度函数。由于这些函数一般都比较复杂，因此，本例[illegible]都将作出必要的解释，帮助读者理解优化求解的方法。

step 1 选择命令窗口菜单栏中的"File"→"New"→"M-file"命令，打开M文件编辑器，在其中输入下面的程序代码

```
function f = shufcn(y)
for j = 1: size(y,1)
    f(j) = 0.0;
    x = y(j,:);
    temp1 = 0;
    temp2 = 0;
    x1 = x(1);
    x2 = x(2);
    for i = 1:5
        temp1 = temp1 + i.*cos((i+1).*x1+i);
        temp2 = temp2 + i.*cos((i+1).*x2+i);
    end
    f(j) = temp1.*temp2;
end
```

上面的程序代码是在本实例中使用的目标函数，将该代码保存为"shufcn.m"文件。

step 2 绘制上面代码文件的函数图形。在MATLAB的命令窗口中输入下面的代码

```
>>plotobjective(@shufcn,[-2 2; -2 2]);
```

输入上面的代码后，按"Enter"键，得到的图形如图4.42所示。

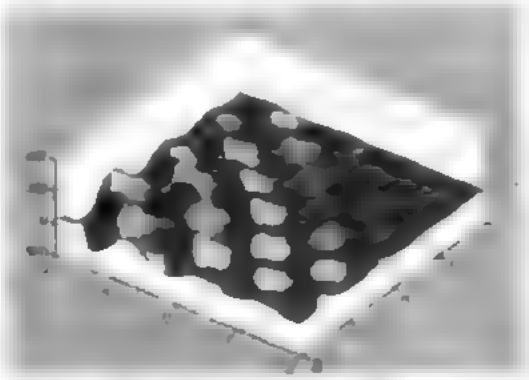


图 4.42 绘制的函数图形

Step 3 将自定义的目标函数作为参数传递给plotobjective函数，[illegible]函数，[illegible]函数的图形，其具体的程序代码如下。

```
function plotobjective(fcn,range)
%PLOTOBJECTIVE plot a fitness function in two dimensions
%   plotObjective(fcn,range) where range is a 2 by 2 matrix in which
each
%   row holds the min and max values to range over in that dimension.

if(nargin == 0)
    fcn = @rastriginsfcn;
    range = [-5,5;-5,5];
end
pts = 100;
span = diff(range')/(pts - 1);
x = range(1,1): span(1) : range(1,2);
y = range(2,1): span(2) : range(2,2);
pop = zeros(pts * pts,2);

k = 1;
for i = 1:pts
    for j = 1:pts
        pop(k,:) = [x(i),y(j)];
        k = k + 1;
    end
end
values = feval(fcn,pop);
values = reshape(values,pts,pts);

surf(x,y,values)
shading interp
light
lighting phong
hold on
contour(x,y,values)
rotate3d
view(37,60)
```

上面程序代码首次使用了绘制曲面的第一个命令，这里又赋予了上直代码的主要含义。首先根据输入的变量值来绘制曲面图，然后绘制出它的等高线。关于具体每个语句的含义，请查看本书中介绍图形绘制的章节。

[illegible]

4.6.2 优化求解

接着上面小节的步骤。

Step 1 设置遗传算法的参数，在MATLAB主命令窗口中输入下面的代码。

```
>> fitnessFunction = @shufcn;
```

[illegible]

STEP 2 [illegible]

```
>>[x,Fval,exitFlag,Output] = ga(fitnessfunction,numberOfVariables);
```

[illegible]

STEP 3 [illegible]

```
[x, fval, reason, output, population, scores] = ga(fitnessfun, nvars,
[illegible]
```

上面命令格式中，各参数的具体含义如下。

- **输入参数**：[illegible] opt fons 为是该优化问题的优化属性。
- **输出参数**：[illegible]。

[illegible]

STEP 4 查看优化求解结果，在 MATLAB 命令窗口输入[illegible]

```
>> fprintf('The best variable value found was : %g,%g\n', x(1),x(2));
>>fprintf('The best function value found was : %g\n', Fval);
>>fprintf('The number of generations was : %d\n', output.generations);
>>fprintf('The number of function evaluations was : %d\n', output
funcCount);
The best variable value found was : -1.44823,-0.791518
The best function value found was : -185.48
The number of generations was : 82
The number of function evaluations was : 1640
```

[illegible]为-185.48。最后，遗传算法的总体代数为82。

[illegible]

4.6.3 设置优化求解的可视化属性

延续上面小节的步骤，

Step 1 [illegible] GAPLOTBESTF 和 GAPLOTSTOPPING，在命令窗口中输入下面的程序代码

```
>>opts = gaoptimset('PlotFcns',{@gaplotbestf,@gaplotstopping});
```

[illegible]

Step 2 [illegible] 函数程序代码，这里仅仅列出函数 gaplotbestf 的程序代码如下

```
function state = gaplotbestf(options,state,flag)
%GAPLOTBESTF Plots the best score and the mean score.
%   STATE = GAPLOTBESTF(OPTIONS,STATE,FLAG) plots the best score as
%[illegible]
    xlabel Generation
    [illegible]
end
hold on;
generation = state.Generation;
best = min(state.Score);
[illegible]
[illegible]
[illegible]
[illegible]
hold off;
```

[illegible] 过程中的各种计算结果属性。

[illegible]

4.6.4 设置遗传算法的属性

继续上面小节的步骤。

step 1 重新运行遗传算法，进行优化求解，在命令窗口中输入如下命令并按回车键：

```
>>[x,fval,exitflag,output,population,score]=ga(fun,2,options);
>>fprintf('The best variable value found was : %g,%g\n', x(1),x(2));
>>fprintf('The best function value found was : %g\n', fval);
>>fprintf('The number of generations was : %d\n', output.generations);
>>fprintf('The number of function evaluations was : %d\n', output.
funccount);
The best variable value found was : -0.799506,-1.42193
The best function value found was : -186.706
The number of generations was : 30
The number of function evaluations was : 600
```

step 2 查看函数求解过程的变化，除了显示上面的优化结果之外，MATLAB还会显示出相应的函数图形。其中，查运行之中的函数图形如图4.43所示。

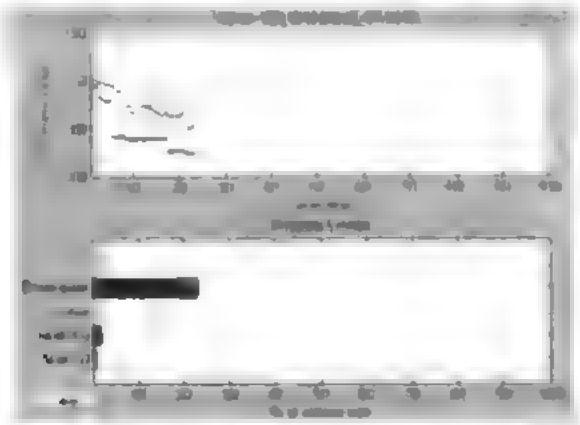


图4.43 运行中的遗传算法属性图形

step 3 查看函数求解结束的图示，当系统结束遗传算法优化的时候，函数图形如图4.44所示。

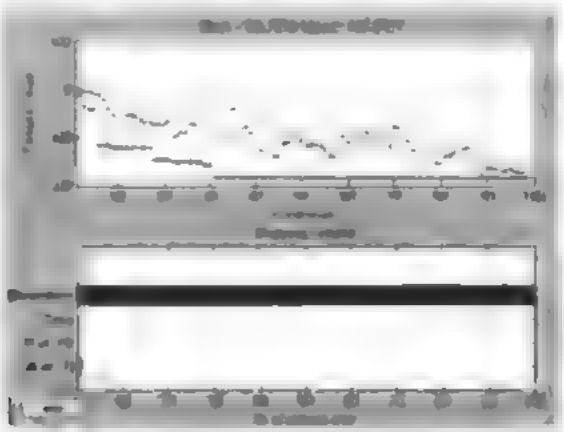


图4.44 运行结束的遗传算法属性图

从上面的图中可以看出，优化求解过程中适应度函数的变化情况，也就是说，最高适应度为"-186.706"，其他的参数设置与前文相同。

465 设置遗传算法的"种群"属性

接续上面小节的步骤。

step 1 [illegible]

```
>>opts = gaoptimset(opts,'PopInitRange',[-1 0;1 2])
```

在上面的程序代码中，使用gaoptimset的典型格式

```
options = gaoptimset(oldopts,'param1',value1,...)
```

[illegible]

初始范围都是从 -1 到 1。

[illegible]

step 2 [illegible]

```
[illegible]
fprintf('The best variable value found was : %g,%g\n', x(1),x(2));
fprintf('The best function value found was : %g\n', Eval);
[illegible]
fprintf('The number of generations was : %d\n', output.generations);
fprintf('The number of function evaluations was : %d\n', output.funccount);
Optimization terminated: stall generations limit exceeded.
The best variable value found was : -0.790095,4.85739
The best function value found was : -185.551
The number of generations was : 60
The number of function evaluations was : 600
```

step 3 [illegible]，如图4.45所示。

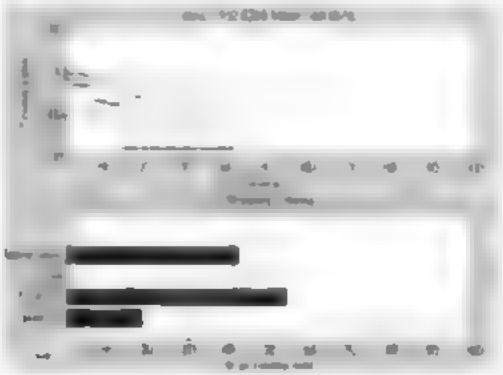


图4.45 运行中的优化属性图形

Step 5 [illegible]

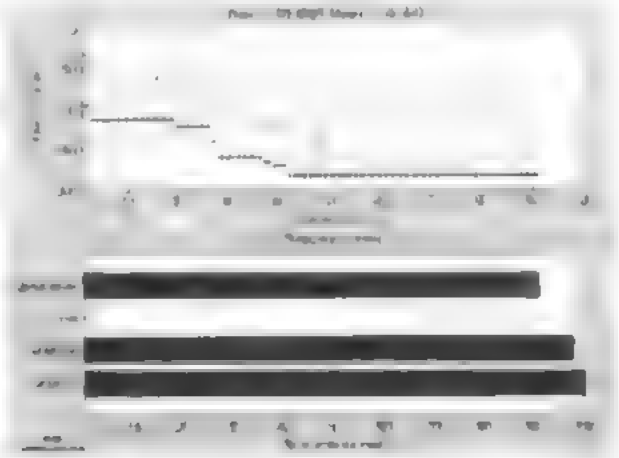


图 4.46 运行后的优化属性图形

[illegible]

4.6.6 设置遗传算法的"中止"属性

[illegible]

Step 1 [illegible]

```
>>opts = gaoptimset(opts,'Generations',250,'StallGenLimit', 50);
```

[illegible]

Step 2 [illegible]

```
>>[x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables,opts);
>>fprintf('The best variable value found was : %g,%g\n', x(1),x(2));
>>fprintf('The best function value found was : %g\n', Fval);
>>fprintf('The number of generations was : %d n', Output.generations);
>>fprintf('The number of function evaluations was : %d\n', Output.
[illegible]
The best variable value found was : -0.818726,-1.51072
The best function value found was : -164.704
The number of generations was : 97
The number of function evaluations was : 1940
```

Step 3 [illegible]

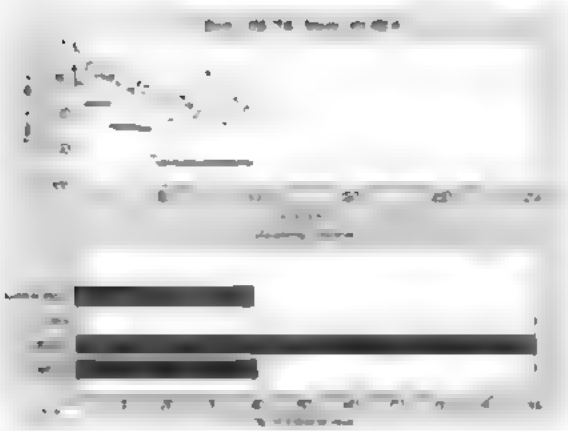


图 4.27 修改属性后的优化曲线

[illegible]

4.7 优化求解综合实例：优化“Banana”函数

[illegible]

由于涉及的步骤比较复杂，下面分小节详细介绍其相应的步骤。

4.7.1 分析目标函数

例 4.25 [illegible]

Step 1 [illegible]

```
>> figure;
[illegible]
>> y=-1:.2:3;
[illegible]
>> zz=100*(yy-xx.^2).^2+(1-xx).^2;
>> surfHndl=surface(x,y,zz,'EdgeColor',[.8 .8 .8]);
>> view(10,55);
>> colormap(hsv);
>> hold on;
>> [c,contHndl]=contour3(x,y,zz+50,[100 500],'k');
>> set(contHndl,'Color',[.8 .8 .8]);
>> drawnow
[illegible]
    'MarkerSize',16,
    [illegible]
    [illegible]
[illegible]
    [illegible]
    [illegible]
```

```
>> plot3(1,1,0,'ko', ...
      'Marker Size',[illegible]
      '[illegible]', [illegible] ...
      'EraseMode','none');
>> text(0.8,1.4,0,'   End', ...
      '[illegible]', [illegible],
      'EraseMode','none');
>> title('Banana Function')
>> grid on
```

Step 2 在banana函数的图形窗口中输入下面的程序代码后，按"Enter"键，[illegible]所示。

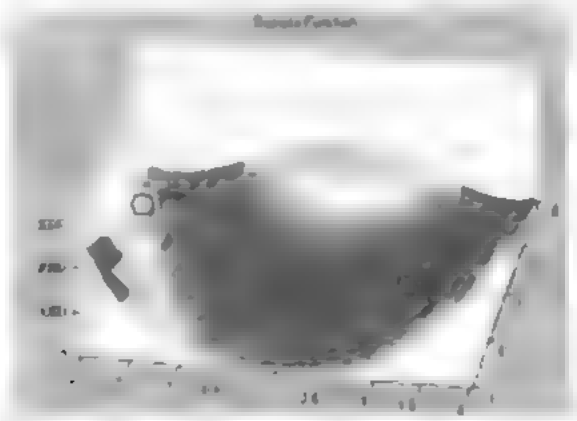


图4.48 绘制目标函数的图形

在上面的图形中，首先绘制了出现函数的曲面图，然后绘制了(1,1,0)为起始点(banana)，终点(1.1,[illegible])为结束点(End)。在上面的绘图命令中使用的"EraseMode"参数，而且在函数的输出中，[illegible]。这样做是为了使这两个点能够在随后的绘图过程中保持可见，以便观察优化算法的进展。

Step 3 编写绘制搜索路径函数。选择文件菜单中的"File"→"New"→"M-file"命令，打开M文件编辑器，在其中输入下面的程序代码

```
function [stop] = banana(x,optimvalues,state,varargin)
%
% Output function that plots the iterates
stop = false;
if strcmpi(state,'iter')
  xpbanplt(x);
end
%xpbanplt函数的程序代码
function out = xpbanplt(currPos,prevPos)
%XPBANPLT Plots one step of solution path.
if nargin==2,
    x1=currPos(1);
    y1=currPos(2);
    z1=100*(y1-x1^2)^2+(1-x1)^2;
    plot3(x1,y1,z1,'g.', ...
        'EraseMode','none',
```

```
        'MarkerSize', [],
    drawnow; % Draws current graph now
    out = [];
[illegible]
    [illegible];
    [illegible];
    [illegible];
    y2=currPos(2);
    z1=[illegible](y1, x1, z1, [illegible]-x1), ...
    z2=[illegible](y2-x2, z2, [illegible]-x2), [illegible];
    plot3([x1 x2],[y1 y2],[z1 z2],'b-', ...
        'EraseMode','none', ...
        'LineWidth',2);
    plot3([x1 x2],[y1 y2],[z1 z2],'g.', ...
        'EraseMode','none', ...
        'MarkerSize',25);
    drawnow; % Draws current graph now
    out = [];
end
```

输入上面的程序代码，保存为名称为"[illegible].m"文件，或将该代码加入要调用在上面制作的目标函数体内，[illegible]

4.7.2 "Broyden Fletcher-Goldfarb-Shanno"优化求解

接续上面小节的步骤，

Step 1 使用"Broyden Fletcher-Goldfarb-Shanno"算法[illegible]，输入下面的程序代码

```
>>x0=[-1.9 2];
>>OPTIONS=[illegible],'-fr','[illegible]',[illegible];
>>GRAD=[illegible];
>> f=[illegible];
>>OPTIONS = [illegible],'MaxFunEvals',[illegible],
            [illegible],[illegible].
>>[x,fval,exitflag,output] = fminunc(f,GRAD,x0,OPTIONS)
```

Step 2 参数优化结束，输入[illegible]，按"Enter"键，[illegible]

```
[illegible]
options.TolFun.
x =
    0.9998    0.9996
fval =
  3.4588e-008
exitflag =
     1
output =
       iterations: 34
        funcCount: 50
         stepsize: 1
    firstorderopt: 5.6450e-004
```

```
[illegible]
    message: [1x53 char]tcnsnh:
```

[illegible]

优化求解结果正确。

[illegible]

step 3 [illegible]

4.44所示。

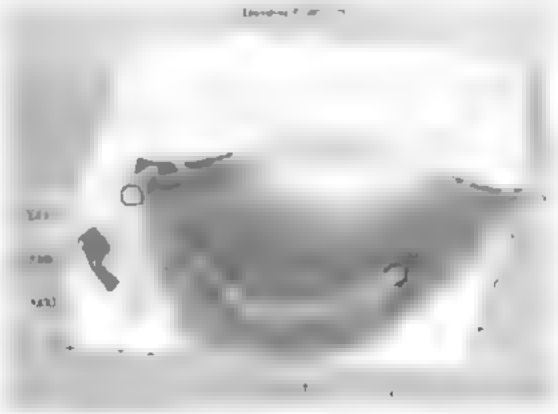


图4.44 优化求解的过程（一）

[illegible]

4.7.3 "Davidon-Fletcher-Powell"优化求解

[illegible]

step 1 [illegible]

程序代码

```
[illegible]
>>OPTIONS = optimset(OPTIONS,'Hessupdate','dfp','gradobj','on',
[illegible]
>> GRAD='[100*(4*x(1)^3-4*x(1)*x(2))+2*x(1)-2; 100*(2*x(2)-2*x(1)^2)]';
>> f='100*(x(2)-x(1)^2)^2+(1-x(1))^2';
>>[x,fval,exitflag,output] = fminunc(f,GRAD,x0,OPTIONS);
```

step 2 [illegible]

```
[illegible] than
 [illegible]
x =
    1.0000    1.0000
fval =
  1.2[illegible]e-009
[illegible]
     1
output =
       [illegible]
       funcCount: 64
        stepsize: 1
   firstorderopt: 0.0011
       [illegible]
         message: [1x85 char]
```

[illegible]的次数为64，得到的结果为[1，1]，表示该优化求解结果正确。

"Davidon-Fletcher-Powell" [illegible] "Broyden-Fletcher-Goldfarb-Shanno" [illegible] 在"Broyden-Fletcher-Goldfarb-Shanno"[illegible]，[illegible]而在"Davidon-Fletcher-Powell"[illegible]，"[illegible]"属性为"[illegible]"。[illegible]

step 3 [illegible]，如图4.50所示。

图4.50 优化求解的过程（二）

[illegible]

4.7.4 "无约束非线性"优化求解

[illegible]

step 1 [illegible]

```
[illegible]
[illegible]
>> [x,fval,exitflag,output] = fminsearch(f,x0,OPTIONS)
```

step 2 [illegible]

```
Exiting: Maximum number of function evaluations has exceeded
         increase MaxFunEvals option.
         Current function value: 0.000000
x =
    0.9999    0.9999
fval =
  5.5109e-009
exitflag =
     0
output =
    iterations: 169
     funcCount: 201
     algorithm: 'Nelder-Mead simplex direct search'
       message: [1x149 char]
```

step 3 [illegible]

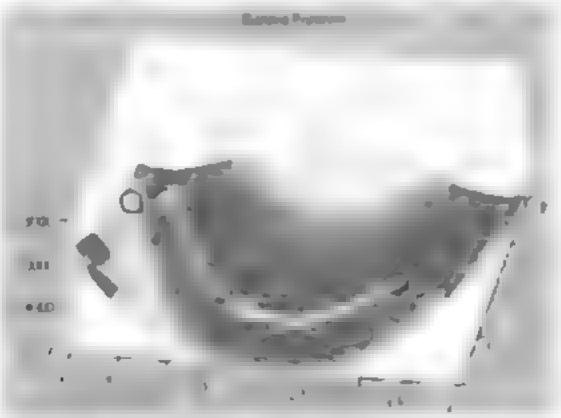


图4.51 优化求解的过程（三）

[illegible]

4.7.5 "最小方差"优化求解

[illegible]按照上面小节的步骤。

step 1 [illegible]中输入下面的程序代码

```
[illegible]
```

```
'lsqnonlin','on'),
>> JAC='[-20*x(1), 10; -1, 0]';
>> f='[10*(x(2)-x(1)^2),(1-x(1))]';
>>[x,resnorm,residual,exitflag,output] = lsqnonlin([f,JAC],x0,[],[],
```

Step 2 查看程序运行结果，在输入[illegible]后按下"Enter"键，程序运行结果如

```
Optimization terminated: search direction less than TolX.
x =
    1.0000    1.0000
[illegible] =
  1.1461e-018
residual =
  1.0e-008 *
    0.1059    0.0165
exitflag =
     4
output =
        iterations: 11
         funcCount: 48
          stepsize: 1.0006
       cgiterations: []
      firstorderopt: []
          algorithm: 'medium-scale: Gauss-Newton, line-search'
            message: 'Optimization terminated: search direction less than
TolX '
```

Step 3 查看优化求解过程的图形。初始值为[illegible]，MATLAB每一步优化求解过程如图4.52所示。

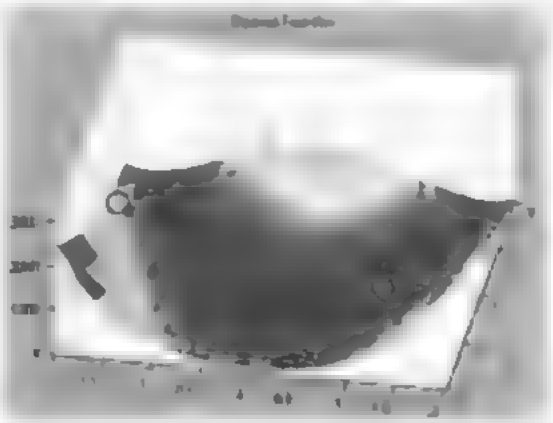


图 4.52 优化求解过程的图形

[illegible]

4.8 优化求解综合实例：复杂的二次规划

[illegible]

[illegible]

自数据和数据点的精度的平方模决定。

[illegible]

4.8.1 设置约束条件

例4.26 使用二次规划的方法来创建帐篷的造形。

step1 [illegible]

```
>>largeL = zeros(36);
>>mask = [6 7 30 31];
>>largeL(mask,mask) = .3*ones(4);
>>largeL(18:19,18:19) = .5*ones(2);
>>xx = [1:5,5:6,6:15,15:16,16:25,25:26,26:30];
>>[XX,YY] = meshgrid(xx);
>>axis([1 30 1 30 0 .5],'off');
>>surface(XX,YY,largeL,'facecolor',[.5 .5 .5],'edgecolor','none');
[illegible]
[illegible]
>>view([-20,30]);
>>title('The set of tent poles')
```

step2 [illegible]

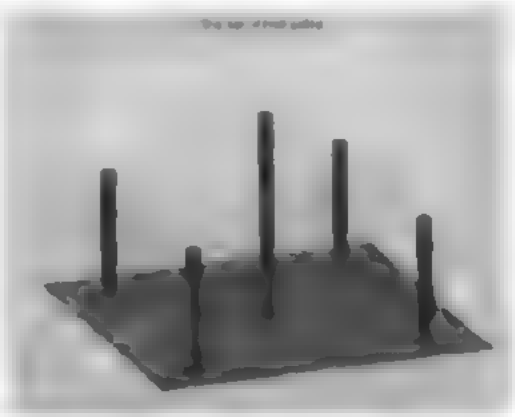


图4.53 绘制帐篷的顶柱

说明 [illegible]

step3 [illegible]

```
% determine a lower bound for the tent
[illegible]
>>E = ones(2);
```

```
>>L(5:6,2
% Add L t
```

step 4 [illegible]

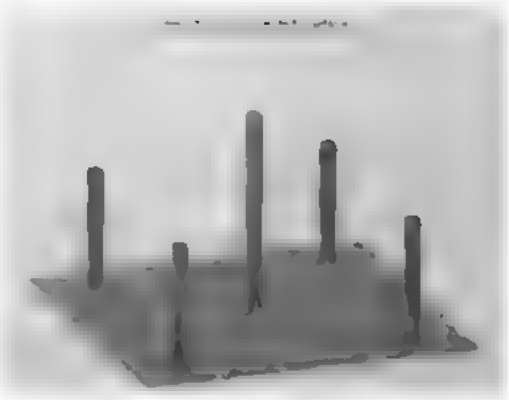


图 4.54 添加下限的曲面

step 5 [illegible]

```
>>sstart = .5*ones(30,30);
% Add it to the plot.
>>surface(sstart,'FaceColor','none','LineStyle','none',
[illegible]
>>title('Initial Value (blue) and Lower Bound (magenta)');
>>set(gcf,'renderer','zbuffer');  % Markers do not show up in OpenGL.
```

step 6 [illegible]

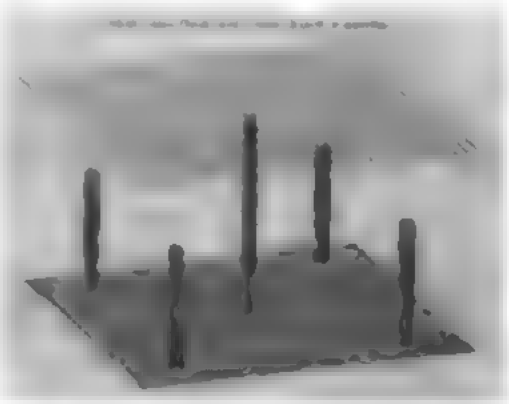


图 4.56 设置优化的初始值表面

step 7 [illegible]

```
>>low = reshape(L,900,1);
>>xstart = reshape(sstart,900,1);
% Illustrate the reordering.
% Draw grid points.
>>xx = 0:4;
>>[X Y] = meshgrid(xx,xx);
>>gpts = plot(X(:),Y(:),'b.');
[illegible]
>>axis off; axis([-2 12 -1.5 5.5]);
>>hold on
>>l(1) = line([7.5 6.5],[2 2.5]);
>>l(2) = line([7.5 6.5],[2 1.5]);
>>l(3) = line([7.5 5.5],[2 2]);
[illegible]
>>yy = 0.2*xx;
>>zz = [-1.5+yy,yy,1.5+yy,3+yy,4.5+yy];
[illegible]
>>set(vect,'markersize',9);
[illegible]
>>hold off;
```

Step 8 查看在[illegible]的结果，在输入上面的程序代码后，按"Enter"键，得到的结果如图4.56所示。

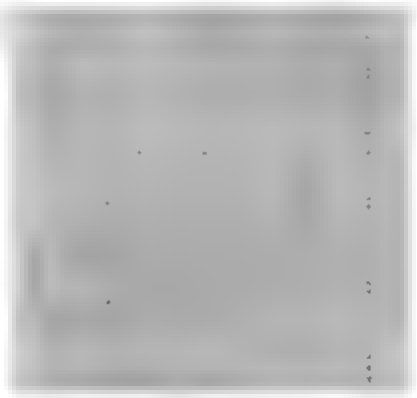


图4.56 形状转换示例

4.6.2 定义目标函数

接续上面小节的步骤。

Step 1 定义目标函数。根据前面的介绍，绘制帐篷的目标函数为

$$\min_{x} \ \frac{1}{2}x^T Hx + c^T x$$

其中，$\frac{1}{2}x^T Hx + c^T x$ 是质量函数的离散拟合，同时，其中的变量满足 $lb \le x$。在MATLAB的命令窗口中输入下面的代码

```
>>H = delsq(numgrid('S',30+2));
>>h = 1/(30-1);
>>c = -h^2*ones(30^2,1);
```

[illegible]

step 2 绘制上面结构，在命令窗口，输入下面代码

```
spy(H);
title('Structure of Hessian Matrix');
```

step 3 查看稀疏结构图。在输入上面的程序代码后，按“Enter”键，可生成稀疏矩阵的结构，如图4.57所示。

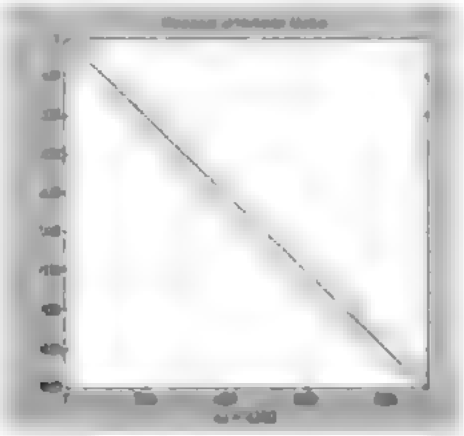


图4.57 稀疏矩阵的结构

4.6.3 进行优化求解

接续上面小节的步骤。

step 1 绘制数据点的相对位置。由于在不断修改的时候需要显示状态窗口，因此在进行优化运算之前，需要首先以样本数据为例绘制其中的相对位置图示。在MATLAB的命令窗口中输入下面的程序代码。

```
>>load testdata;
>>plot(xxx1,xx1,'b.',xxx2,xx2,'r.');
>>set(gca,'YTick',[-1 1]);
>>set(gca,'YTickLabel',{'lower';'upper'});
>>axis([1 900 -1 1]);
>>title('Relative position of x(i) to upper and lower bounds (log-scale)')
```

step 2 查看绘制结果。输入上面代码后，按“Enter”键，得到的图形如图4.58所示。

[illegible]

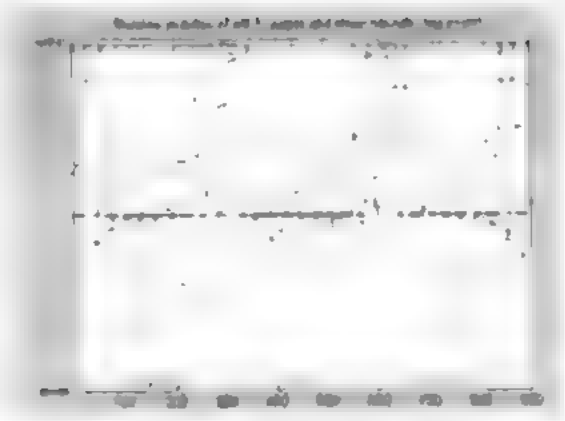


图 4.58 各个数据点的相对坐标

step 3 [illegible]

```
>>currplot = plot(xGG2,GG2,'b.',xGG1,GG1,'r.');
>>title('Relative gradient scaled to the range -1 to 1');
>>legend('abs(grad) > tol','abs(grad) <= tol',4);
>>axis([1 900 -1 1]);
[illegible]
[illegible]
```

step 4 [illegible]

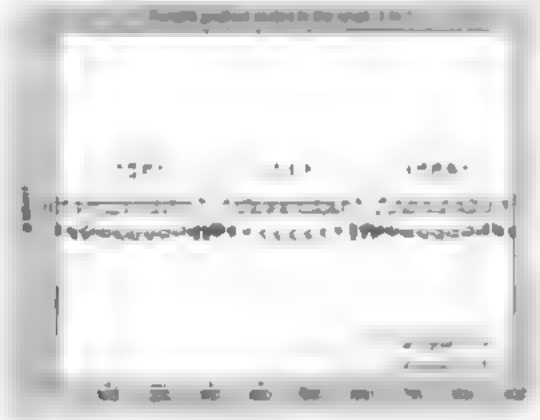


图 4.59 各个数据点的相对单位梯度图形

说明 [illegible]

step 5 设置优化属性，进行优化求解。在命令窗口中输入下面的程序代码

```
>>options = optimset('LargeScale','on','display','off', ...
   'ShowStatusWindow','iterplus');
>>x = quadprog(H, c, [], [], [], [], low, [], xstart, options);
```

step 6 查看优化求解过程的图形，在上面的程序代码中，首先设置要显示关于优化求解的状态窗口，然后在默认条件下，进行二次规划求解。在命令窗口中输入 [illegible]

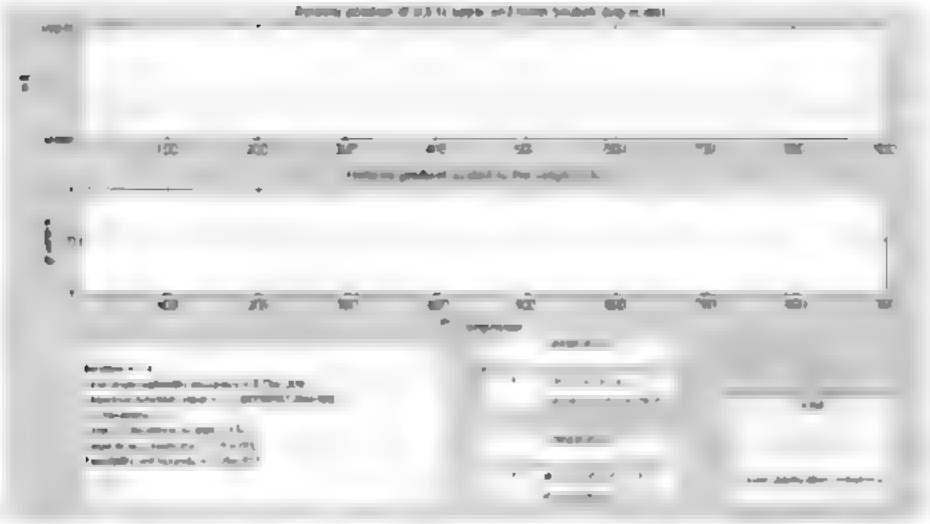


图 4.60 显示迭代过程信息

[illegible]

step 1 [illegible]

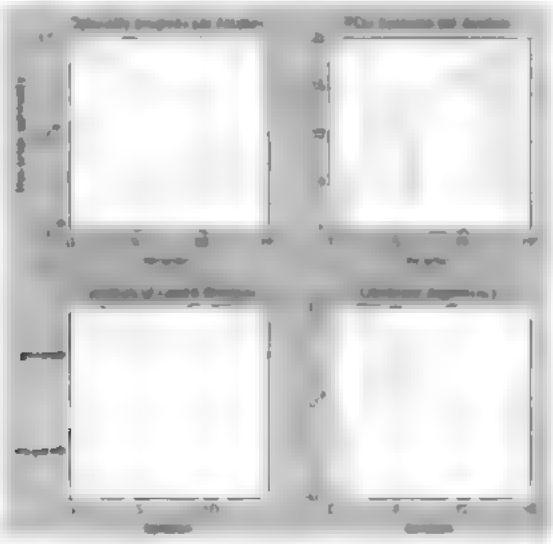


图 4.61 算法运行属性

[illegible]

4.8.4 绘制优化求解的结果

继续上面小节的步骤。

Step 1 利用优化结果绘制曲面图。在命令窗口中输入下面的程序代码

```
>>S = reshape(x,30,30);
% Close figures that QUADPROG creates (if they are still open).
[illegible]
[illegible]
[illegible]
[illegible]
>>surf(L,'facecolor',[ .5 .5 .5]);
[illegible]
[illegible]
>>axis off
>>axis tight;
>>view([ -20,30] );
[illegible]
>>surf(L,'facecolor',[ .5 .5 .5]);
>>surface(L,'edgecolor','k','facecolor','none');
>>title('Solution Surface')
>>axis off
>>axis tight;
>>view([ -20,30] );
```

Step 2 查看程序代码的结果。在输入上面代码后，按“Enter”键，得到的图形如图4.62所示。

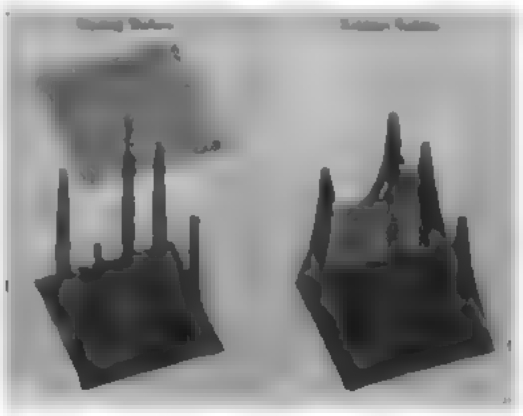


图4.62 绘制求解的曲面图

Step 3 设置外箱表面的属性。在命令窗口中输入下面程序代码

```
[illegible]
>>surf(L,'facecolor',[ 0 0 0] );
[illegible]
[illegible]
>>hold off;
>>axis tight;
>>axis off;
>>view([ -20,30] );
```

Step 4 查看代码的结果。输入上面代码后，按“Enter”键，得到的图形如图4.63所示。

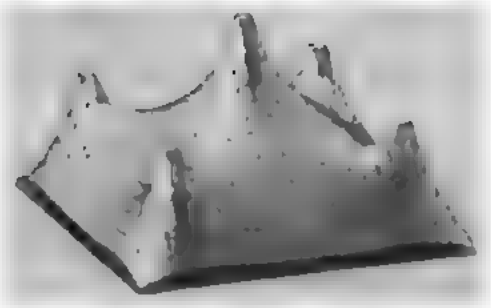


图 4.63 绘制的帐篷

4.9 常微分方程

[illegible]

4.9.1 显性常微分方程

在数学理论中，显性常微分方程具有下面的形式

$$y'=f(t,y),\ y(t_0)=y_0$$

[illegible]

件，微分方程需要求解的是 $y(t)$。

[illegible]

多个命令，表 4.2 列出了各命令的使用情况。

表 4.2 常微分方程组的解法

命令	常用的算法	精度	应用
ode45	[illegible]	中等	[illegible]改变结果
ode23	[illegible]	低	[illegible]ode45好
ode113	[illegible]	中等	[illegible]情况
ode15s	[illegible]	[illegible]	[illegible]
ode23s	[illegible]	低	[illegible]时候
ode23t	[illegible]	低	[illegible]
ode23tb	TR－BDF2 方法	低	对于误差容许范围比较宽的情况

[illegible]

[illegible]

性方程组。

[illegible]

- [illegible] ode45、ode23和ode113
- 非刚性方程组：ode45、ode23和ode113

[illegible]

（例如ode45），具体的调用格式如下：

- [t,Y] = solver(odefun,tspan,y0,options)
- [t,Y,TE,YE,IE] = solver(odefun,tspan,y0,options)

在上述的调用格式中，各参数的具体含义如下：

- 参数"odefun"表示ODE函数的名称
- 参数"tspan"在两种[illegible]
- 参数y0则表示微分方程的初始数值
- 参数options[illegible]，options的具体数值可以由函数odeset来获得
- 参数t是所求数值解的自变量数据列向量
- 参数Y表示所求微分方程的因变量数据矩阵
- 参数[illegible]

[illegible]

例4.27 使用MATLAB的命令来求解非刚性方程组的数值解

$$\begin{cases} y_1' = y_2 y_3 \\ y_2' = -y_1 y_3 \\ y_3' = -0.51 y_1 y_2 \end{cases}$$

其中初值条件为：$y_1(0)=0$，$y_2(0)=1$和$y_3(0)=1$。

Step 1 [illegible]中输入下面的程序代码：

```
function dydt = euler(t,y)
dydt = [      y(2)*y(3)
             -y(1)*y(3)
          -0.51*y(1)*y(2) ];
```

[illegible]

Step 2 在 MATLAB 命令窗口中输入如下命令：

```
%定义微分方程求解的时间范围
>> tspan = [0 12];
%定义微分方程的初值条件
>>y0 = [0; 1; 1];
%进行微分方程求解
[illegible]
%绘制微分方程组的计算结果
[illegible]
>>hold on
[illegible]
>>plot(t,Y(:,3),'b.')
%设置绘制图形的属性
>>axis([0 12 -1.2 1.2])
[illegible]
>>title('The Numerical Solution of Euler Equation')
```

Step 3 命令执行完毕后，在弹出的绘图窗口中，按 Enter 键，即可得到如图 4.64 所示的结果。

图 4.64 微分方程的数值解

[illegible]

- [illegible]
- [illegible]
- [illegible]

在 MATLAB 中求解微分方程组时，为了提高计算精度，有时也需要编程解决。为了提高微分方程组的求解速度，还可以使用 MATLAB 的内置函数 [illegible]，来提高编程的效率。

例 4.28 使用 MATLAB 的内置函数求解一阶微分方程组的数值解。

$$\begin{cases} y_1' = -0.04y_1 + 10^4 y_2 y_3 \\ y_2' = -0.04y_1 - 10^4 y_2 y_3 - 3\times 10^7 y_2^2 \\ y_3' = 3\times 10^7 y_2^2 \end{cases}$$

其中初值条件为：$y_1(0)=0$，$y_2(0)=0$ 和 $y_3(0)=0$。

step 1 选择命令窗口编辑栏中的菜单"File"→"New"→"M-file"命令，打开M文件编辑器，在其中输入下面的程序代码

```
function dydt = stiffex1(t,y)
dydt = [  (-0.04*y(1) + 1e4*y(2)*y(3))
          (0.04*y(1) - 1e4*y(2)*y(3) - 3*10^7*y(2)^2)
          (3*10^7*y(2)^2)];
```

在输入程序代码后，保存该文件为"stiffex1.m"文件，该函数还需要一个特殊参数。

step 2 在MATLAB命令窗口中输入下面命令

```
>> tspan=[0 4*logspace(-6,6)];
>> y0 = [1; 0; 0];
>> [t,y] = ode15s(@stiffex1,tspan,y0);
>> y(:,2) = 1e4*y(:,2);
>> semilogx(t,y(:,1),'r-','LineWidth',1.5);hold on;
>> semilogx(t,y(:,2),'g.','LineWidth',1.5);hold on;
>> semilogx(t,y(:,3),'b.');
>> ylabel('1e4 * y(:,2)');
>> axis([10^(-10) 10^10 -0.1 1.1])
>> legend('y1','y2','y3')
>> set(gca,'Ytick',[-0.1:0.1:1.1])
>> title('Stiff Equation solved by ODE15S');
>> grid
```

step 3 在命令窗口中输入下面命令，按"Enter"键，绘制刚性微分方程的数值解。

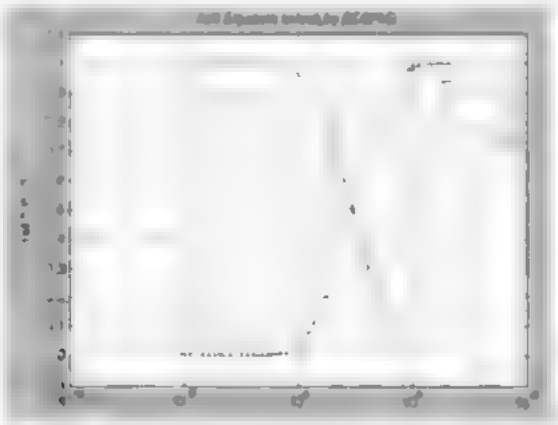


图 4.65 刚性微分方程的数值解

说明 在上面的本例中，由于其中的数值解的标准值很小，因此属于难解方程。需要使用ode15s命令进行求解，可以用计算机进行实际应用 ode45 命令求解的结果。

4.9.2 设置允许误差属性

[illegible]

[illegible]

在对[illegible]属性[illegible]，在 MATLAB 中，[illegible]解[illegible]参数的命令为 odeset 函数，其调用的调用格式如下

- options = odeset('name1',value1,'name2',value2,...)：用参数名称和参数数值来分别定义解法器的各种参数
- [illegible] = odeset(oldopts,'name1',value1,...)：修改命令[illegible]解法器 oldopts，改变的是指定参数的数值。

在设置解法器的属性参数之前，有几种方法可以查看各参数的默认数值，可以使用 odeset 命令来查看所有属性参数默认数值

从上面可知，在[illegible]解法器参数的[illegible]，其中，[illegible]'Error Tolerances'参数如表 4.3 所示。

表 4.3 允许误差的属性参数

属性名称	取值	默认数值	说明
RelTol	标量	1e-3	用于[illegible]误差必须小于相对误差和数值解的乘积并且小于绝对误差
AbsTol	标量或向量	1e-6	确定误差一定的绝对值，如果是标量，则表示误差适用于所有的分量。如果是向量则单独指定每一个分量的绝对误差
[illegible]	on或者off	off	如果该属性数值为on，[illegible]误差控制精度，如果该属性数值为off，则采用更加严格的精度控制计算

关于解[illegible]算法处理[illegible]种复杂，需要了解外[illegible]，对于可[illegible]问题，设置不同的误差属性，会直接影响[illegible]解的[illegible]和效率

例 4.29 设置[illegible]属性，[illegible]

Step 1 在 MATLAB 命令窗口中输入下[illegible]

```
>> options = odeset('RelTol',1e-4,'AbsTol',[1e-4 1e-4 1e-5]);
[illegible]
>> [t1,Y1] = ode45(@euler,[0 12],[0 1 1],options);
[illegible]
>>tic;
>> [t,Y] = ode45(@euler,[0 12],[0 1 1]);
[illegible]
```

Step 2 查看[illegible]，在输入[illegible]

```
%设置属性后消耗的时间
>> time_set
time_set =
[illegible]
%默认属性下使用的时间
>> time_default
time_default =
    0.0300
```

[illegible]

Step 3 [illegible] MATLAB [illegible] 输入下面的命令

```
%设置属性和默认属性条件下的时间间隔
>> size_set=size(t1,1)
size_default=size(t,1)
size_set
    85
size_default =
```

[illegible]

Step 4 [illegible]

```
%定义微分方程求解的时间区间
>> tspan = [0 12];
%定义微分方程的初值条件
>>y0 = [0; 1; 1];
%进行微分方程求解
>> [t1,Y1] = ode45(@euler,tspan,y0,options);
%绘制微分方程组的计算结果
[illegible]
>>hold on;
[illegible]
[illegible]
%设置绘制图形的属性
>>axis([0 12 -1.2 1.2])
[illegible]
>>grid on
>>title('The Numerical Solution of Euler Equation')
```

Step 5 [illegible]

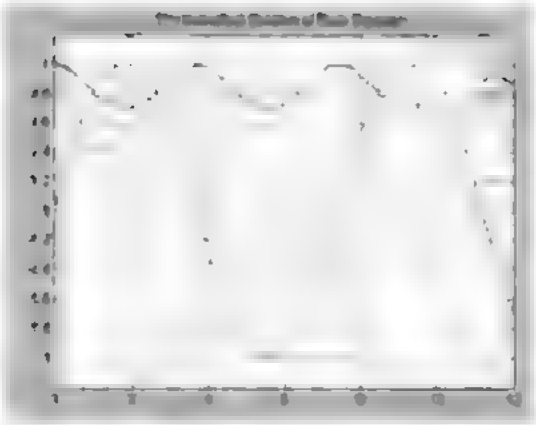


图 4.66 设置属性后的图形

[illegible]

4.9.3 设置输出参数属性

在 MATLAB 中，解法器输出（solver Output），参数如表 4.4 所示。

表 4.4 解法器输出的属性参数

属性名称	取值	默认数值	说明
[illegible]	字符串	odeplot（[illegible]）	[illegible]，使用此函数输出结果。在调用ode类的函数时，如果有输出变量，在默认情况下将使用函数odeplot绘制结果，如果没有输出变量，则不输出结果
[illegible]	数组变量	[illegible]	[illegible]输出函数OutputFcn中输出。默认情况下，将输出所有的下标
[illegible]	逻辑数	[illegible]	[illegible]，[illegible]会返回值
[illegible]	字符串	[illegible]	[illegible]，输出计算结果[illegible]

在参数"OutputFcn"中，除了可以使用odeplot命令[illegible]odiffmss与odeprint等函数命令，也可，自行编写用户自定义函数。

例 4.30 设置解法器输出的属性，查看求解例 4.26 中的洛伦兹吸引子方程组。

step 1 使用默认属性。在 MATLAB 的命令窗口中输入下面的命令

```
>> tspan = [0 12];
y0 = [0; 1; 1];
figure;
ode45(@euler,tspan,y0);
```

step 2 查看[illegible]的结果，在输入[illegible]，按"Enter"键，得到[illegible]

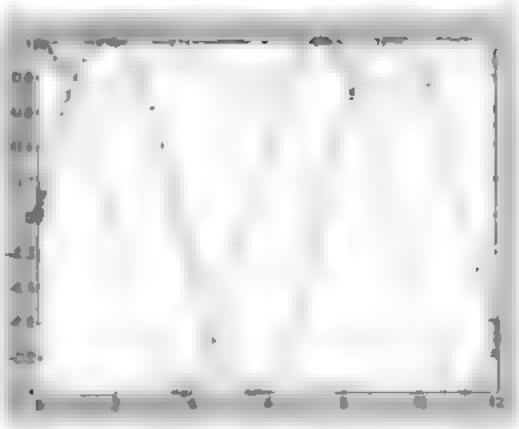


图 4.67 默认输出条件下的图形

step 3 重新设置输出属性，并求解微分方程。在命令窗口中输入下面的命令

```
>> options=odeset('OutputFcn',@odephas3,'Stats','on');
>> figure;
>> ode45(@euler,tspan,y0,options);
```

step 4 查看程序的运行结果。在输入上面的命令后，按“Enter”键，得到的结果如图 4.68 所示

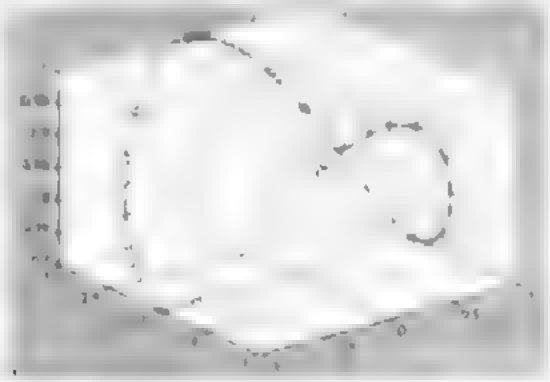


图 4.68 修改输出属性后的图形

时，在 MATLAB 的命令窗口中显示求解的各种信息

```
19 successful steps
2 failed attempts
127 function evaluations
```

如果不涉及不完全或部分的函数，可以使用默认的求解器进行修改：在默认绘图属性的基础上，进行之前的修改，使其能够输出函数；自定义默认的绘制曲线；使用绘图命令进行设置或编辑

step 5 查看默认函数的程序代码。如果希望使用第三种方法输出函数，首先需要查看默认函数的程序代码，以函数“odeplot”为例，其默认存储路径为..MATLAB7.0 toolbox\matlab\funfun，同时其程序代码如下

```
function status = odeplot(t,y,flag,varargin)
status = 0;                          % Assume stop button wasn't pushed.
```

```
chunk = 128;                                % Memory is allocated in chunks.
if nargin < 3 || isempty(flag) % odeplot(t,y) [v5 syntax] or odeplot(t,
y,'')
  ud = get(gcf,'UserData');
  % Append t and y to ud.t and ud.y, allocating if necessary.
  nt = length(t);
  chunk = max(chunk,nt);
  [rows,cols] = size(ud.y);
  oldi = ud.i;
  newi = oldi + nt;
  if newi > rows
    ud.t = [ud.t; zeros(chunk,1)];
    ud.y = [ud.y; zeros(chunk,cols)];
  end
  ud.t(oldi+1:newi) = t;
  ud.y(oldi+1:newi,:) = y.';
  ud.i = newi;
  set(gcf,'UserData',ud);
  if ud.stop == 1                           % Has stop button been pushed?
    status = 1;
  else
    ylim = get(gca,'ylim');
    % Replot everything if out of axis range or if just initialized.
    if (oldi == 1) || (min(y(:)) < ylim(1)) || (ylim(2) < max(y(:)))
      for j = 1:cols
        set(ud.lines(j),'Xdata',ud.t(1:newi),'Ydata',ud.y(1:newi,j));
      end
    else
      % Plot only the new data.
      for j = 1:cols
       set(ud.line(j),'Xdata',ud.t(oldi:newi),'Ydata',ud.y(oldi:newi,j));
      end
    end
  end
  else
  switch(flag)
  case 'init'                               % odeplot(tspan,y0,'init')
    ud = [];
    cols = length(y);
    ud.t = zeros(chunk,1);
    ud.y = zeros(chunk,cols);
    ud.i = 1;
    ud.t(1) = t(1);
    ud.y(1,:) = y.';
    f = figure(gcf);
    if ~ishold
      ud.lines = plot(ud.t(1),ud.y(1,:),'-o');
      hold on
      ud.line = plot(ud.t(1),ud.y(1,:),'-o','EraseMode','none');
      hold off
      set(gca,'XLim',[min(t) max(t)]);
    else
      ud.lines = plot(ud.t(1),ud.y(1,:),'-o','EraseMode','none');
      ud.line = plot(ud.t(1),ud.y(1,:),'-o','EraseMode','none');
    end
    % The STOP button.
```

```
        h = findobj(f,'Tag','stop');
        if isempty(h)
          ud.stop = 0;
          pos = get(0,'DefaultUicontrolPosition');
          pos(1) = pos(1) - 15;
          pos(2) = pos(2) - 15;
          str = 'ud=get(gcf,''UserData''); ud.stop=1; set(gcf,''UserData'',ud);';
          uicontrol( ...
              'Style','push', ...
              'Units','pixel', ...
              'Position',pos, ...
              'Callback',str, ...
              'Tag','stop');
        else
          set(h,'Visible','on');              % make sure it's visible
          if ishold
            oud = get(f,'UserData');
            ud.stop = oud.stop;               % don't change old ud.stop status
          else
            ud.stop = 0;
          end
        end
        set(f,'UserData',ud);
         case 'done'                          % odeplot([],[],'done')
        f = gcf;
        ud = get(f,'UserData');
        ud.t = ud.t(1:ud.i);
        ud.y = ud.y(1:ud.i,:);
        set(f,'UserData',ud);
        cols = size(ud.y,2);
        for j = 1:cols
          set(ud.lines(j),'Xdata',ud.t,'Ydata',ud.y(:,j));
        end
        if ~ishold
          set(findobj(f,'Tag','stop'),'Visible','off');
          set(gca,'xLimMode','auto');
          refresh;                        % redraw figure to remove marker frags
        end
          end
    end
    drawnow;
```

说明

[illegible]

494 设置解法器其他属性

关于解法器的其他属性，这里就不详细介绍了，感兴趣的读者可以查阅MATLAB中关于帮助文件。下面选择一个比较综合的例子说明如何设置属性，来求解具体的问题。

例4.31 求解小球反弹的数学模型，该小球在受到完全弹性碰撞后为下落弹性的情况，使用微分方程来求解小球运动的轨迹。

step 1 选择命令窗口编辑栏中的"File" ⇨ "New" ⇨ "M File"命令，打开M文件编辑器，在其中输入下面的程序代码：

```
function dydt =bounce (t,y)
dydt = [ y(2); -9.8];
```

在输入上面的程序代码后，将该程序代码保存为"bounce.m"文件。

step 2 选择命令窗口编辑栏中的"File" ⇨ "New" ⇨ "M-File"命令，打开M文件编辑器，在其中输入下面的程序代码：

```
function [ value,isterminal,direction] = events(t,y)
% Locate the time when height passes through zero in a decreasing direction
% and stop integration.
value = y(1);      % detect height = 0
isterminal = 1;    % stop the integration
direction = -1;    % negative direc
```

输入上面的程序代码后，将该程序代码保存为"events.m"文件，该函数为该微分方程的求解过程中的"定位事件"函数，这个函数将在ode相关命令运行时被调用。

step 3 进行微分方程的求解。在MATLAB的命令窗口中输入下面的代码：

```
%设置微分方程的时间跨度和初始数值
>>tstart = 0;
>>tfinal = 30;
>>y0 = [ 0; 20];
%设置进行数据插值
refine = 4;
%设置微分方程的属性
options = odeset('Events',@events,'OutputFcn',@odeplot,'OutputSel',1,...
                 'Refine',refine);
%创建新的图形窗口，并设置对应的属性
figure;
set(gca,'xlim',[ 0 30],'ylim',[ 0 22]);
box on
hold on;
tout = tstart;
yout = y0.';
teout = [];
yeout = [];
ieout = [];
for i = 1:10
  % 执行第一次微分方程的求解
  [ t,y,te,ye,ie] = ode23(@bounce,[ tstart tfinal],y0,options);
  if ~ishold
    hold on
  end
  % 累积输出结果
  nt = length(t);
  tout = [ tout; t(2:nt)];
  yout = [ yout; y(2:nt,:)];
  teout = [ teout; te];
  yeout = [ yeout; ye];
  ieout = [ ieout; ie];
```

```
    ud = get(gcf,'UserData');
    if ud.stop
      break;
    end
      % 设置新的积分条件
    y0(1) = 0;
    y0(2) = -.9*y(nt,2);
    % 设置时间步参数的数值
    % 使用了 'refine' 属性值 4
    options = odeset(options,'InitialStep',t(nt)-t(nt-refine),...
                             'Max[illegible]',[illegible]);
    tstart = t(nt);
[illegible]
[illegible]
% 设置图形的属性
[illegible]
[illegible]
title('Ball trajectory and the events');
hold off
% 调用函数来绘制图形
[illegible]
[illegible]
[illegible]
```

STEP 6 单击按钮[illegible]，在编辑[illegible]，按[illegible]键，[illegible]图4.68所示。

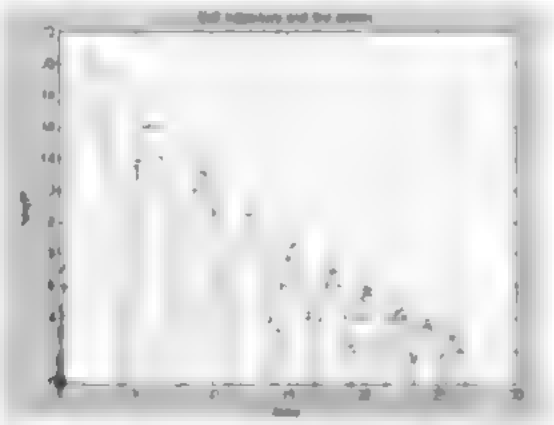


图4.69 小球运动轨迹图形

4.9.5 加权常微分方程

加权常微分方程的通用形式如下

$$M(t,y)y'=f(t,y)\quad y(t_0)=y_0$$

在[illegible]，[illegible]，[illegible]，[illegible]，并且[illegible]，[illegible]。在MATLAB中，求解加权常微分方程的步骤大致如下：

- 编写加权函数 $M(t,y)$ 的M文件
- 通过odeset命令设置微分方程的"mass"属性

◆ 选择合适的 ode 类函数来求解微分方程。

在本小节中，将以一个比较简单的例子来说明如何求解加权常微分方程。

例 4.32 求解下面的加权常微分方程，其中

$$M(t,y)=\begin{pmatrix}1 & 0 & 0 & 0 & 0 & 0\\ 1 & m_1+m_2 & 0 & 0 & 0 & -m_2L\sin(y(5))\\ 1 & 0 & 0 & 0 & 0 & 0\\ 1 & 0 & 0 & m_1+m_2 & 0 & m_2L\cos(y(5))\\ 1 & 0 & 0 & 0 & 0 & 0\\ 1 & -L\sin(y(5)) & 0 & L\cos(y(5)) & 0 & L^2\end{pmatrix}$$

同时，微分方程函数如下：

$$f(t,y)=\begin{pmatrix}y(2)\\ m_2L\cdot y^2(6)\cos(y(5))\\ y(4)\\ m_2L\cdot y^2(6)\sin(y(5))-(m_1+m_2)g\\ y(6)\\ -gL\cos(y(5))\end{pmatrix}$$

根据上面的微分方程的参数矩阵，其对应的微分方程组如下：

$$\begin{cases}y(1)=X\\ y(2)=X'\\ y(3)=Y\\ y(4)=Y'\\ y(5)=\theta\\ y(6)=\theta'\end{cases}$$

在上面的方程组中，(X,Y)表示的是物体的坐标数值，θ表示的是物体运动的角度，同时对应的初始数值为 y_0=[1;4;2;20;pi/2;2]。

step 1 选择命令窗口编辑栏中的"File" ⇨ "New" ⇨ "M-File"命令，打开 M 文件编辑器，在其中输入下面的程序代码：

```
function M = mass(t,y,m1,m2,L,g)
% 创建新的矩阵，开辟矩阵存储空间
M = zeros(6,6);
M(1,1) = 1;
M(2,2) = m1 + m2;
M(2,6) = -m2*L*sin(y(5));
M(3,3) = 1;
M(4,4) = m1 + m2;
M(4,6) = m2*L*cos(y(5));
M(5,5) = 1;
M(6,2) = -L*sin(y(5));
M(6,4) = L*cos(y(5));
M(6,6) = L^2;
```

在输入上面的程序代码后，将其保存为"mass.m"文件，该文件将是在后面的步骤中调用的

加权函数。

step 2 选择命令窗口编辑栏中的“File”→“New”→“M-File”命令，打开M文件编辑器，在其中输入下面的程序代码

```
function dydt = massode(t,y,m1,m2,L,g)
dydt = [
  y(2)
  m2*L*y(6)^2*cos(y(5))
  y(4)
  m2*L*y(6)^2*sin(y(5))-(m1+m2)*g
  y(6)
  -g*cos(y(5))/L
  ]
```

在输入上述程序代码后，将其保存为“massode.m”文件，该函数为函数名为massode的微分方程组的要求解的微分方程组。

说明 在上面的程序代码中，初始时刻的参数m_1、m_2、L和g，这些参数都需要在后面主程序中定义具体的数值。

step 3 在MATLAB的命令窗口中输入下面的程序代码

```
%定义微分方程的参数
m1 = 0.1;
m2 = 0.1;
L = 1;
g = 9.81;
%设置微分方程的属性
tspan = linspace(0,4,25);
y0 = [ 0; 4; 2; 20; -pi/2; 2];
options = odeset('Mass',@mass);
%进行微分方程的求解
[ t y] = ode45(@odemass,tspan,y0,options,m1,m2,L,g);
%定义绘制的变量数值
theta = y(1,5);
X = y(1,1);
Y = y(1,3);
xvals = [ X X+L*cos(theta)];
yvals = [ Y Y+L*sin(theta)];
%绘制变量的曲形
figure;
plot(xvals,yvals,xvals(1),yvals(1),'r.',xvals(2),yvals(2),'g.')
title 'A three-body problem with mass matrix M(t,y), solution y(t)'!;
axis([ 0 22 0 24])
hold on
for j = 2:length(t)
  theta = y(j,5);
  X = y(j,1);
  Y = y(j,3);
  xvals = [ X X+L*cos(theta)];
  yvals = [ Y Y+L*sin(theta)];
  plot(xvals,yvals,xvals(1),yvals(1),'r.',xvals(2),yvals(2),'g.')
```

```
end
set(gca,'YLim',[0 24]);
set(gca,'Ytick',[0:2:24]);
grid on
hold off
```

step 5 [illegible]

图 4.70 微分方程的数值解

说明 [illegible]

4.9.6 延迟微分方程命令

在数学理论中，延迟微分方程具有下面的形式

$$y(t)=f(t,y(t),y(t-\tau_1),\cdots,y(t-\tau_n))$$

[illegible]

在 MATLAB 中，[illegible]

```
sol = dde23(ddefun,lags,history,tspan,options)
```

上面的命令中各参数的具体含义如下。

- **ddefun**：[illegible]
- **lags**：[illegible]
- **history**：[illegible]
- **tspan**：[illegible]
- **options**：[illegible]

[illegible]

4.9.7 延迟微分方程实例

例 4.33 求解下面的延迟微分方程组

$$\begin{cases} y_1(t) = y_1(t-1) \\ y_2(t) = y_1(t-1) + y_2(t-0.2) \\ y_3(t) = y_3(t) \end{cases}$$

[illegible]，[illegible]$y_1(t)=y_2(t)=y_3(t)=1$，其中 $t \le 0$，求解区间 [illegible] $[0,5]$，滞后的数据量为 $[1,0.2]$。

step 1 [illegible]开 M 文件编辑器，在其中输入下面的程序代码

```
function dydt = ddex1de(t,y,Z)
% Differential equations function for DDEX1.
ylag1 = Z(:,1);
ylag2 = Z(:,2);
dydt = [ ylag1(1)
         ylag1(1) + ylag2(2)
         y(2)                ];
```

[illegible]的微分方程组。

step 2 [illegible] "New" [illegible] 命令，[illegible] M 文件编辑器，在其中输入下面的程序代码

```
function s = ddex1hist(t)
% Constant history function for DDEX1.
s = ones(3,1);
```

[illegible]的历史函数。

step 3 [illegible] MATLAB [illegible]输入下面的代码

```
sol = dde23(@ddex1de,[1, 0.2],@ddex1hist,[0, 5]);
[illegible]
plot(sol.x,sol.y,'LineWidth',1.5)
title('An example of Wille'' and Baker.');
xlabel('time t');
ylabel('solution y');
grid
legend('y1','y2','y3')
```

step 4 [illegible]，在输入[illegible]，按 "Enter" 键，[illegible]

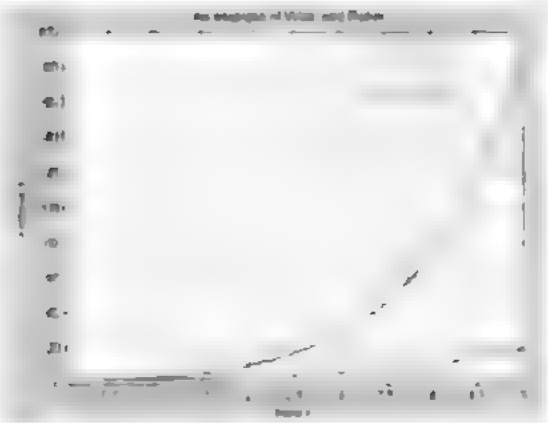


图4.7 微分方程的数值解

在方程的求解过程中，通过[illegible]的数值，[illegible]命令绘制出[illegible]结果图形。

4.9.8 常微分方程的边界问题

[illegible]种基本的方法。

- **迭加法**：[illegible]问题。
- **试射法**：[illegible]值问题。
- **松弛法**：[illegible]。

在MATLAB中，[illegible]

$$\frac{dy}{dx}=f(x,y)$$

[illegible]需要求解y在[a,b]中进行数值求解。

[illegible]

$$\frac{dy}{dx}=f(x,y,p)$$

[illegible]

在MATLAB中求解微分方程的相关命令如下

- [illegible]
- [illegible]
- [illegible]

下面详细介绍上面各命令的参数含义。

- [illegible]
- [illegible]
- [illegible]

[illegible]

4.9.9 边界问题实例

例 4.34 求解下面的微分方程

$$y'' + \frac{2}{x}y' + y^3 = 0$$

同时该方程的边界条件为 $y'(0)=0$，$y(1)=\frac{\sqrt{3}}{2}$。

根据相关的分析，该常微分方程有解析解，表达式为 $f(x)=\frac{1}{\sqrt{1+\frac{x^2}{3}}}$，[illegible]

step 1 编写微分方程的代码。选择菜单栏中的 "File" → "New" → "M-file" 命令，打开 M 文件编辑器，在其中输入下面的程序代码

```
function dydx = emdenode(x,y)
%EMDENODE  Evaluate the function f(x,y)
dydx = [   y(2)
           -2/x*y(2) - y(1)^3 ];
```

在输入上面的程序代码后，将其保存为 "emdenode.m" 文件，该文件将是在后面步骤中调用的微分方程组。

step 2 编写边界条件函数的代码。选择菜单栏中的 "File" → "New" → "M-file" 命令，打开 M 文件编辑器，在其中输入下面的程序代码

```
function res = emdenbc(ya,yb)
%EMDENBC  Evaluate the residual in the boundary conditions
res = [  ya(2)
         yb(1) - sqrt(3)/2 ];
```

在输入上面的程序代码后，将其保存为 "emdenbc.m" 文件，该文件将是在后面步骤中调用的边界条件函数。

Step 3 [illegible]

```
% 设置微分方程与解
S = [0  0
     0 -2];
options = bvpset('SingularTerm',S);
% This constant guess satisfies the boundary conditions.
guess = [sqrt(3)/2; 0];
solinit = bvpinit(linspace(0,1,5),guess);
sol = bvp4c(@emdenode,@emdenbc,solinit,options);
% The analytical solution for this problem.
x = linspace(0,1);
truy = 1 ./ sqrt(1 + (x.^2)/3);
% 绘制微分计算的结果
[illegible]
[illegible]
[illegible]
plot(sol.x,sol.y(1,:),'ro');
% 设置图形其他属性
title('Emden problem -- BVP with singular term.')
[illegible]
[illegible]
ylabel('solution y');
[illegible]
```

Step 4 [illegible]

图 4.72 微分方程求解的图形效果

说明 在上面的结果中，[illegible]

例 4.35 求解下面的微分方程

$$y''+|y|=0$$

同时该方程满足边界条件为 y(0)=0 和 y(4)=-2。

Step 1 [illegible] 开 M 文件编辑器，在其中输入下面的程序代码

```
function dydx = twoode(x,y)
%TWOODE  Evaluate the differential equations for TWOBVP.
dydx = [  y(2); -abs(y(1)) ];
```

在输入上面的程序代码后，将其保存为"twoode.m"文件，该文件将是在后面步骤中调用的微分方程组。

step 2 编写边界条件函数的代码。选择命令窗口编辑栏中的"File" ⇨ "New" ⇨ "M-File"命令，打开M文件编辑器，在其中输入下面的程序代码：

```
function res = twobc(ya,yb)
%TWOBC  Evaluate the residual in the boundary conditions for TWOBVP.
res = [  ya(1); yb(1) + 2 ];
```

在输入上面的程序代码后，将其保存为"twobc.m"文件，该文件将是在后面步骤中调用的边界条件函数。

step 3 求解微分方程。在MATLAB的命令窗口中输入下面的代码：

```
% One solution is obtained using an initial guess of y1(x)=1, y2(x)=0
solinit = bvpinit(linspace(0,4,5),[1 0]);
sol = bvp4c(@twoode,@twobc,solinit);
x = linspace(0,4);
y1 = deval(sol,x);
figure;
plot(x,y1(1,:),'r','LineWidth',1.5);
xlabel('x');
ylabel('y');
set(gca,'Ytick',[-2:0.4:2.4])
grid on
title('The first solution')
% The other solution is obtained using an initial guess of y1(x)=-1, y2
(x)=0
solinit = bvpinit(linspace(0,4,5),[-1 0]);
sol = bvp4c(@twoode,@twobc,solinit);
y2 = deval(sol,x);
% Plot both solutions
figure;
plot(x,y1(1,:),'r','LineWidth',1.5);
hold on;
plot(x,y2(1,:),'g','LineWidth',1.5);
set(gca,'Ytick',[-2:0.4:2.4])
xlabel('x');
ylabel('solution y');
title('A BVP with two solutions');
grid on;
```

step 4 查看程序代码的结果。在输入上面的程序代码后，按"Enter"键，得到的第一个数值结果如图4.73所示。

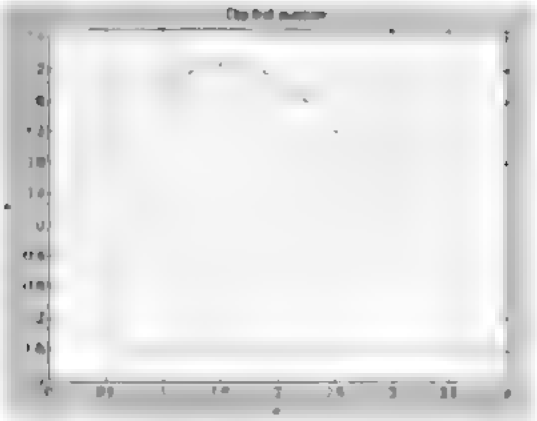


图 4.73 第一个数值求解的结果

由于该微分方程有两个数值解，两个数值解的结果如图 4.74 所示。

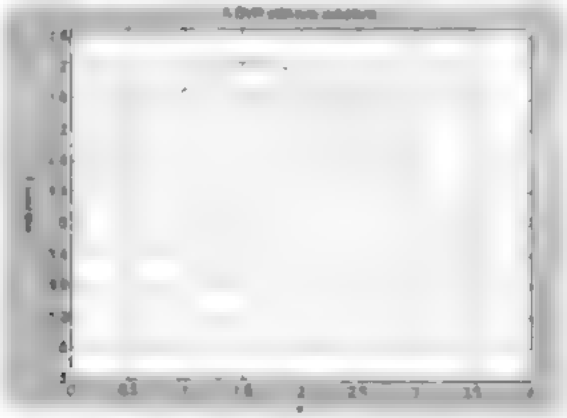


图 4.74 两个数值解的图形

从上面的结果应可以看出，在对微分方程组求解时，解出了两个高阶微分方程的数值解。

4.10 小结

在本章中，笔者介绍了一些使用 MATLAB 来进行统计分析的数学方法，数据插值、曲线拟合、傅里叶分析、统计建模方法等。这些方法在前面章节中为读者介绍，要知道的，这些的数学原理方法都不入，笔者建议读者在学习本章之前先了解，能够对这个数学原理一起理解。在后面的章节中，将介绍如何使用 MATLAB 进行符号计算。

◆ ◆ ◆

第5章 符号计算

本章包括

- 符号对象
- 符号函数
- 符号代数方程组
- 符号积分变换
- 符号表达式
- 符号微积分
- 符号微分方程
- 利用 Maple

在前面的章节中，已经介绍MATLAB在数值计算和分析中的应用，MATLAB除了能够处理数值运算之外，还可以进行各种符号计算。在MATLAB中，进行符号计算可以用推理解析的方式进行，避免数值计算带来的截断误差，同时符号计算可以得到正确的封闭解后者精确的数值解。而且，在MATLAB7.0中调用符号计算的命令十分简单，和读者在教科书中使用的公式符号大体相同，可以十分容易地被接受。

在MATLAB中，符号数学工具箱（Symbolic Math Toolbox）中的工具都是建立在数学计算软件Maple的基础上的。如果在MATLAB7.0中进行符号运算，MATLAB会调用MAPLE进行运算然后将结果返回到MATLAB7.0的命令窗口中。正是因为这个原因，当MATLAB7.0进行软件升级或者符号计算内核MAPLE的升级时，符号计算工具箱也随之升级，这些升级内容对于普通用户来说，差别还是比较细微的，对于有比较大变化的地方，在本章的相关位置也会分别给出说明。

在MATLAB中，符号运算实质上属于数值计算的补充部分，并不能算是MATLAB的核心内容。但是，关于符号计算的命令、符号计算结果的图形显示、计算程序的编写或者帮助系统等，都是十分完整和便捷的。

符号对象和符号表达式

在科学工程中，数值运算是十分重要的内容，但是自然科学理论中，各种公式、表达式以及相应的推导等也是十分重要的，这些就是符号运算解决的重点内容。在MATLAB中，数值和数值变量用于数值的存储和各种计算，而符号对象、变量、函数以及相应的操作都是用来形成符号表达式的，然后按照相关数学内容的规则进行运算，得出相应的解析解。

sym 命令

和在MATLAB中使用数值计算一样，使用数值表达式的变量必须首先进行变量赋值，否则MATLAB会返回变量错误信息。符号数学工具箱也沿用该规则，在进行符号运算之前，首先需要定义符号对象，然后利用这些符号对象去构建表达式，最后才能进行符号运算。

从上面的介绍中可以看出，创建符号对象是进行符号运算的基础，MATLAB提供多种创建符号对象的命令。数值、字符串、符号对象是MATLAB中常见的三种变量，MATLAB提供将数值或者字符串变量转化为符号对象的方法，同时提供将符号对象转换成为数值或者字符串变量的方法。

本小节首先介绍如何创建符号对象，关于其他的转换方法将在后面小节中详细介绍。定义符号

对象的常见命令是sym，常见的调用格式如下：

```
Sym1=sym(argn,flagn)
Sym2=sym('argv',flagv)
```

在上面的调用格式中，使用sym命令创建了符号对象Sym1和Sym2。下面详细介绍这两种调用格式的含义。

Sym1=sym(argn,flagn) 将数值或者数值表达式转换为符号对象Sym1，参数flagn的作用是定义转换的符号对象应该符合的格式，其具体的选项和含义如下：

- 'd' 用最接近的十进制浮点精确表示；
- 'e' 当表示数值计算时，带估计误差的有理表示；
- 'f' 用十六进制浮点表示；
- 'r' 这是MATLAB的默认设置，用最接近有理表示的形式。

表达式Sym2=sym('argv',flagv)将指定的字符串变量argv转换为符号对象Sym2，参数flagv的作用也是定义转换的符号对象应该符合的格式，其具体的选项和含义如下：

- 'positive' 限定A表示为正的实型符号变量；
- 'real' 限定A为实型符号变量；
- 'unreal' 限定A为非实型符号变量。

使用sym命令创建符号对象

例5.1 将数值或者数值表达式转化为符号对象，使用不同的转换格式。

step 1 在MATLAB的命令窗口中输入下列内容：

```
>> sym1=sym([ 3/4,log(10),exp(2),log(10)+exp(2)],'d');
>> sym2=sym([ 3/4,log(10),exp(2),log(10)+exp(2)],'e');
>> sym3=sym([ 3/4,log(10),exp(2),log(10)+exp(2)],'f');
>> sym4=sym([ 3/4,log(10),exp(2),log(10)+exp(2)],'r');
>> my_sym=[ sym1;sym2;sym3;sym4]
```

step 2 按"Enter"键，就会得到上面程序代码的结果。

```
my_sym =

[ .75000000000000000000000000000000, 2.3025850929940459010936137929093,
7.3890560989306504069418224389665, 9.6916411919246954198570165317506]
[                                3/4,          5184960683398422*2^(-51),
8319337573440942*2^(-50),          5455908957570076*2^( 49)]
[          '1.8000000000000'*2^(-1),           '1.26bb1bbb55516'*2^(1),
'1.d8e64b8d4ddae'*2^(2),           '1.3621ecb57c41c'*2^(3)]
[                                3/4,          5184960683398422*2^(-51),
8319337573440942*2^(-50),          5455908957570076*2^(-49)]
```

在上面的程序中，分别将数值3/4，数值表达式log(10)、exp(2)等转换为符号变量，差别在于不同的转换格式，根据上面的结果可以看出转换格式对结果的影响。

step 3 查看程序结果的信息。在上面的结果中，sym1～sym4都是符号变量，而my_sym则是符号

[illegible]

```
>> whos
  Name          Size                    Bytes  Class
  [illegible]
  sym1          1x4                       560  sym object
  sym2          1x4                       454  sym object
  sym3          1x4                       490  sym object
  sym4          1x4                       454  sym object
Grand total is 782 elements using 3740 bytes
```

[illegible]

例5.2 [illegible] $\int_a^b \frac{1}{x}dx = \ln\frac{b}{a}$

STEP 1 [illegible]

```
>> Var=sym('var','positive');          %定义止的积分变量
>>Upper=sym('upper','real');            %定义积分上限
>>Lower=sym('lower','real');            %定义积分下限
>>Integral= int(1/(Var),Lower,Upper)    %计算积分数值
```

STEP 2 [illegible]

```
Integral =
[illegible]
```

[illegible] $\log(upper) - \log(lower) = \ln(upper) - \ln(lower) = \ln\frac{upper}{lower}$ [illegible] $\int_{lower}^{upper}\frac{1}{x}dx = \ln\frac{upper}{lower}$，[illegible] $\int_a^b \frac{1}{x}dx = \ln\frac{b}{a}$

[illegible] 为real。

[illegible]

例5.3 [illegible]

STEP 1 [illegible]

```
>> s1=sym('[ 3/7,exp(1),exp(2)+log(4)]');
>> s2=sym('[ 3/7 exp(1) exp(2)+log(4)]');
>> s3=sym([ 3/7,exp(1),exp(2)+log(4)]);
>> sd= s1;s2;s3];
```

step 2 [illegible]

```
>> g1
ad =
[                3/7,                exp(1),        exp(2)*log(4)]
[                3/7,                exp(1),        exp(2)*log(4)]
[                3/7, 6121026514868074*2^(-51), 4940083132741141*2^(-49)]
```

[illegible]是绝对准确的。

[illegible]比较节省资源

5.1.3 使用syms命令创建符号对象

[illegible]

在MATLAB中，syms命令的常见调用格式如下

```
syms arg1 arg2 ... flagv
```

[illegible]表示转换格式，和sym命令相同。

例5.4 使用syms命令创建符号变量。

step 1 [illegible]

```
>> clear all
>> syms alpha beta thro real;
>> whos
```

step 2 [illegible]

```
>> whos
  Name        Size                    Bytes  Class
  alpha       1x1                       134  sym object
  beta        1x1                       132  sym object
  thro        1x1                       132  sym object
Grand total is 16 elements using 398 bytes
```

[illegible]解这种命令。[illegible]

```
>> alpha=sym('alpha','real');
>> beta=sym('beta','real');
```

[illegible]

[illegible]令，这符合 MATLAB 符号运算简洁的特点。

在上面的程序代码中，第一行代码 [illegible] 使用 syms 命令创建符号变量

[illegible]

- 变量名可以由英文字母、数字或者下划线组成
- 变量名必须由英文字母开始
- 组成变量名称的字符长度不能超过 31。
- MATLAB 会区别变量名称的大小写。

5.1.4 符号计算的运算符和函数

前面 [illegible] 符号对象，但是，[illegible]

[illegible] MATLAB 编程上带来极大的方便。

下面简单介绍符号计算中的运算符和函数的使用方法

- 基础运算符

  [illegible]
- 关系运算符

  [illegible]
- 三角、双曲函数

  在 MATLAB 中，[illegible]）及对应的反函数，都可以用在符号运算中。
- 指数、对数函数

  在 MATLAB 中，[illegible]，只使用 log 函数，[illegible]
- 复数函数

  在 MATLAB 中，[illegible]，在符号运算中，没有提供求相角的函数。
- 矩阵代数命令

  在 MATLAB 中，[illegible] 异解的 svd 命令有所不同，将在后面章节中详细介绍。

[illegible]

5.1.5 识别对象的命令

[illegible]，MATLAB 提供了识别不同对象类型的命令。

[illegible]的对象。下面将使用一个新的例子来说明如何使用这些命令。

例 5.5 使用 MATLAB 命令来识别不同的对象类型。

Step 1 在 MATLAB 的命令窗口中输入下列内容

```
>> clear all
[illegible]
>> my_char='[ pi,sqrt(5),exp(4)]';
>> my_sym=sym(my_char);
```

[illegible]出命令来识别不同的对象。

Step 2 [illegible]

```
>> Var_class=char(class(my_number),class(my_char),class(my_sym))
Var_class =
[illegible]
[illegible]
[illegible]
```

[illegible] double（双精度）、char（字符串）、sym（符号对象）。

[illegible]

Step 3 [illegible]

```
>> Isa_var=[ isa(my_number,'double'),isa(my_char,'char'),isa(my_sym,
'sym')];
>> Isa_var
Isa_var =

     1     1     1
```

[illegible]

my_sym 是符号（sym）类型。

step 4 使用 whos 命令来判断不同的对象类型，在命令窗口中输入下列内容：

```
>> whos my_char my_number my_sym
  Name            Size                    Bytes  Class
  my_char         1x19                       38  char array
  my_number       1x3                        24  double array
  my_sym          1x3                       274  sym object
Grand total is 40 elements using 336 bytes
```

在上面的程序中，之所以输入的命令是 "whos my_char my_number my_sym"，而不是直接输入命令 "whos"，是因为在前面的步骤中创建了其他变量；为了简化查询结果，就直接查询上面三个变量的类型。

确定符号表达式中的变量

为了简化符号对象的操作和计算，MATLAB 为用户提供了 findsym 命令，可以实现对符号表达式中符号变量或者指定数目的变量的自定认定。

其常见的调用格式如下：

- r = findsym(S) 确定符号表达式或者矩阵 S 中自由符号变量；
- r = findsym(S,n) 确定符号表达式或者矩阵 S 中靠 x 最近的 n 个独立符号变量。

例 5.6 使用 MATLAB 的命令来确定符号表达式中的变量。

step 1 在 MATLAB 的命令窗口中输入下列内容：

```
>> syms a x y z t
```

step 2 确定下面简单符号表达式中的符号变量信息：

```
>> findsym(sin(pi*t))
ans =
t
```

step 3 确定下面简单符号表达式中的符号变量信息：

```
>> findsym(x+i*y-j*z)
ans =
x, y, z
```

step 4 确定下面简单符号表达式中的符号变量信息：

```
>> findsym(a+y,1)
ans =
y
```

从上面的简单实例中可以看到 findsym 函数的使用方法及其使用范围，在后面的使用步骤中可以灵活使用该命令。

符号精度计算

前面已经介绍符号计算的一个重要特点就是计算过程中不会出现舍入误差，可以得到任意精度的计算。也就是说，如果希望计算结果精确，那么就应该牺牲计算时间和存储空间，使用符号计算来得到足够高的计算精度。

在MATLAB的符号计算工具箱中，提供了如下三种不同类型的计算精度。

- **数值类型**：MATLAB7.0中的浮点算术计算。
- **有理数类型**：MAPLE中的精确符号计算。
- **VPA类型**：MAPLE的任意精度算术计算。

上面三种不同的运算方式各有利弊，读者需要在使用过程中根据计算精度、时间和存储空间的要求，选用不同的计算精度。

例5.7 在MATLAB中，使用不同的精度计算数学表达式。

step 1 在MATLAB的命令窗口中输入下列内容：

```
>> format long
>>var_float=1/3+1/5;
>>var_char=sym(1/3+1/5);
```

step 1 在MATLAB的命令窗口中依次输入var_float、var_char查看计算结果如下：

```
>> var_float
var_float =
   0.53333333333333
>> var_char
var_char =
 8/15
```

step 1 使用whos命令查看不同结果的数据类型，得到的结果如下：

```
>> whos var_float var_char
  Name            Size                    Bytes  Class
  var_char        1x1                       132  sym object
  var_float       1x1                         8  double array
Grand total is 6 elements using 140 bytes
```

从上面的运算结果中可以看出，浮点运算结果var_float并不精确，但是计算所占的内存最小，而且运算速度最快。在MATLAB7.0中，双精度输出的数字位数由format命令控制。在上面的运算过程中，存在着三种计算误差：第一种误差来自于由计算1/3的除法舍入误差，第二种误差来自于相加得到的舍入误差，第三种误差来自于二进制转换为十进制的结果。

而另外一面，符号运算结果var_char则十分精确，但是所需要的时间和占用的内存资源都是很大的。而且，通过最后的命令可以看出，符号计算得出的结果都是字符串，尽管从形式上是数值，但是在变量类型上还是字符串。

如果需要从精确解中获得任意精度的解，并改变默认的精度，把任意精度符号解变成数值解，需要使用MATLAB7.0提供了命令：Vpa和digits，其相应命令的调用格式如下：

- digits(d)　设置今后数值计算以d位相对精度进行。
- digits　显示当前采用的数值计算精度。
- R = vpa(A)　在当前设定的精度下，求出A的数值型符号表达式R。
- R = vpa(A,d)　在d位相对精度下，求出A的数值型符号表达式R。

[illegible]

例5.8　在MATLAB中，显示不同的显示精度。

Step 1 在MATLAB命令窗口中输入下列命令。

```
>>  [illegible]
>>  [illegible]
>>  [illegible]
>> v2=vpa(abs(s1-s2),64);
```

Step 2 在MATLAB中输入[illegible]，按[illegible]键，查看[illegible]计算结果。

```
>> [ s1;s2;v1;v2]
 ans =
                                                [illegible]
                                   7781555297764672*2^(-47)
                                     .10099691278601084e-14
 .10099691278601084471695958717297432391994727397e-14
```

Step 3 最后，在MATLAB中输入[illegible]，按[illegible]键，查看[illegible]

```
>> [illegible]
 Digits = 32
```

[illegible]

5.3 符号表达式的操作

从前面的介绍中，读者可以为符号运算建立一个初步的概念。[illegible]，MATLAB为用户提供了许多符号表达式和函数的操作命令，例如表达式分解、展开、简化等，这些命令都可以调用符号表达式的子函数，这些命令使用户能够完成符号表达式的结构查询。MATLAB提供了多个函数(collect、expand、factor、horner等，[illegible])，可以调用符号表达式的[illegible]。下面将分别详细介绍这些函数的用法和应用。

5.3.1 Collect函数

在MATLAB中，collect函数的调用格式如下

- R = collect(S) 将表达式 S 中相同次幂的项合并，S 可以是表达式或者符号矩阵。
- R = collect(S,v) 将表达式 S 中 v 的相同次幂项合并，v 的默认值是 x。

该函数实现的功能是符号表达式中的同类项合并。

例5.9 在MATLAB中，按照不同的方式合并表达式$(x+e^{-y}x^3-y)*(\sqrt{x}y+e^{-2y}x)$中的参数同类项。

step 1 在 MATLAB 的命令窗口中输入下列内容：

```
>> Exper=sym('(x+exp(-y)*x^3-y)*(sqrt(x)*y+exp(-2*y)*x)');
>> R1=collect(Exper,x);
>> R2=collect(Exper,y);
>> R3=collect(Exper,exp(-y));
```

step 2 在 MATLAB 中输入“R=[R1;R2;R3]”后，按“Enter”键，查看使用不同方式合并的结果如下：

```
>> R=[R1;R2;R3]
 R =
  exp(-y)*exp(-2*y)*x^4+exp(-y)*y*x^(7/2)+exp(-2*y)*x^2+y*x^(3/2)
  -y*exp(-2*y)*x-x^(1/2)*y^2-x^(1/2)*y^2+((x+exp(-y)*x^3)*x^(1/2)
  -exp(-2*y)*x)*y+(x+exp(-y)*x^3)*exp(-2*y)*x
  x^3*(x^(1/2)*y+exp(-2*y)*x)*exp(-y)+(x-y)*(x^(1/2)*y+exp(-2*y)*x)
```

step 3 在 MATLAB 中输入“RM=collect(Exper)”后，查看 MATLAB 默认情况下的合并类型，得到的结果如下：

```
>> RM=collect(Exper)
 RM =

exp(-y)*exp(-2*y)*x^4+exp(-y)*y*x^(7/2)+exp(-2*y)*x^2+y*x^(3/2)
-y*exp(-2*y)*x-x^(1/2)*y^2
```

step 4 重新输入表达式，将 Exper 表达式中的符号改为 w、t，查看 MATLAB 合并的不同的结果，得到的结果如下：

```
>> Exper1=sym('(w+exp(-t)*w^3-t)*(sqrt(w)*t+exp(-2*t)*w)');
>> R4=collect(Exper1)
 R4 =

exp(-t)*exp(-2*t)*w^4+exp(-t)*t*w^(7/2)+exp(-2*t)*w^2+t*w^(3/2)
-t*exp(-2*t)*w-t^2*w^(1/2)
```

step 5 分析上面的计算结果：

从上面的步骤（2）中的结果可以看出，当使用不同的合并条件时，同样的符号表达式会得出不同的结果，因此在实际应用中应该根据需要选择不同的合并条件；

从上面的步骤（3）中的结果可以看出，MATLAB 的默认合并条件是按照变量 x 进行合并的，因此 RM 和 R1 得到的表达式完全相同；

从步骤（4）中的结果可以看出，当表达式中没有变量 x 的时候，MATLAB 会按照首先出现的符号变量进行同类项合并。

[illegible]

5.3.2 Expand 函数

在 MATLAB 中，expand 函数主要用于将[illegible]，其语法格式为：

```
R = expand(S)
```

[illegible]

例 5.10 在 MATLAB 中，使用 expand 函数[illegible]

Step 1 在 MATLAB 的命令窗口输入下面内容：

```
>> syms alpha theta a b x y
>> R1=expand(tan(alpha+theta));
>> R2=expand(2^(x+y));
>> R3=expand((a+b)^7);
```

Step 2 在 MATLAB 中输入 "[illegible]"，按 "Enter" 键，查看[illegible]下：

```
>> R=[R1;R2;R3]
R =
                    (tan(alpha)+tan(theta))/(1-tan(alpha)*tan(theta))
                                                                  [illegible]
[illegible]
```

从上面的运行结果，MATLAB 中的 expand 函数[illegible]

其中，前面两个等式分别来自于下面的恒等式

$$\tan(\alpha+\beta)=\frac{\tan\alpha+\tan\beta}{1-\tan\alpha\tan\beta}$$

$$2^{x+y}=2^x\times 2^y$$

[illegible]

5.3.3 Factor 函数

在 MATLAB 中，factor 函数主要是将[illegible]

```
[illegible]
```

其中，S 是多项式或者多项式矩阵，系数是有理数，MATLAB 会将表达式中每一项系数为有理数的低阶多项式相乘的形式。如果多项式 S 不能进行有理数范围内的解，函数会返回 S 本身。

例5.11 在MATLAB中，[illegible]，其中n取[illegible]，8]范围内的整数。

Step 1 在MATLAB的命令窗口中输入下列命令

```
>> syms x y real
>> n=1:8;
>> p=x.^n-y.^n;
>> f=[[illegible]]';
```

Step 2 在MATLAB的命令窗口中输入[illegible]，[illegible]

```
f =

[                    x-y,                    x-y]
[                    [illegible],                    [illegible]]
[                  x^3-y^3,          (x-y)*(x^2+x*y+y^2)]
[                  [illegible],                    [illegible]]
[                  [illegible],                    [illegible]]
[                  [illegible],                    [illegible]]
[                  [illegible],                    [illegible]]
[                  [illegible],                    [illegible]]
```

[illegible]是在实数范围内都可以继续分解。

[illegible]

例5.12 在MATLAB中求[illegible]整数和 $f(n)=10^n-1$ 进行因数分解，其中n取[illegible]，9]范围内的整数。

Step 1 在MATLAB的命令窗口中输入下列内容

```
[illegible]
>> n=1:9;
>> p=sym(10.^n-1);
>> f=conj([p;factor(p)]');
```

Step 2 在MATLAB的命令窗口中输入[illegible]，[illegible]

```
>> f
  f =
[                    9,                    [3, 3^2]]
[                   99,            [3, 3^2*(11)]]
[                  999,            [3, 3^3*(37)]]
[                 9999,       [3, 3^2*(11)*(101)]]
[                99999,       [3, 3^2*(41)*(271)]]
[               999999,  [3, 3^3*(7)*(11)*(13)*(37)]]
[              9999999,       [3, 3^2*(239)*(4649)]]
[             99999999,  [3, 3^2*(11)*(73)*(101)*(137)]]
[            999999999,            [3, 3^4*(37)*(333667)]]
```

在一定的符号标示，使用factor命令求解一个整数的质因子数的解结果，最后，在程序代码中则使用了conj命令求得复数的共轭。

实际上，并非所有的多项式都可以factor并得到整数上的因子，在symbolic Math Toolbox中实现，而是基于特殊的多项式工具箱（MATLAB Function）中实现，但是通常也都会使用factor函数。当然，当涉及多项式，MATLAB会调用数学函数工具箱（MATLAB Function）中的factor函数。

5.3.4 Horner 函数

在MATLAB中，horner函数主要用于将多项式转换为嵌套形式，调用格式为：

```
R = horner(P)
```

其中，P是符号多项式矩阵，函数horner将P中的每个元素转换为嵌套形式，并返回一个

例5.13 在MATLAB中演示对多项式的嵌套分解

step 1 在MATLAB的命令窗口中输入下列内容

```
>> syms t;
>> p=t^8-7*t^7+5*t^6-10*t^5+8*t^4-4*t^3+3*t^2+5*t-8;
>> r=horner(p);
```

step 2 在MATLAB的命令窗口中输入[illegible]，然后按[illegible]键，得到[illegible]

```
>> r
r =

-8+(5+(3+(-4+(8+(-10+(5+(-7+t)*t)*t)*t)*t)*t)*t)*t
```

horner函数可以使用嵌套形式的表达式解结果，它可以将多项式中的每个符号变量的嵌套形式，如果多项式没有符号解，则需要使用[illegible]命令。

例5.14 在MATLAB中演示对多变量的多项式的嵌套分解。

step 1 在MATLAB的命令窗口中输入下列内容

```
>> syms x y;
>> p1=x^2+x+y^3-2*y+x*y-x^2*y^2;
>> p2=x^2+x+y^3-2*y+x*y-x^2*y^2;
>> r1=horner(p1); r2=horner(p2);
>> R=[r1;r2];
```

step 2 在MATLAB的命令窗口中输入[illegible]，然后按[illegible]键，得到[illegible]

```
>> R
R =

[illegible]
[illegible]
```

[illegible]

5.3.5 Numden 函数

[illegible]

[illegible]

例 5.15 [illegible] $\frac{x^2-1}{x+2}+\frac{2x+5}{3x-2}$ [illegible]

step 1 在 MATLAB 的命令窗口中输入下面命令：

```
>> syms x
[illegible]
>> [n,d]=numden(A)
```

step 2 查看运算的结果。按“Enter”键，就可以得到相应的结果如下：

```
[illegible]

[illegible]x+1]

d =

(x+2)*(3*x-2)
```

step 3 [illegible]

$$\frac{x^2-1}{x+2}+\frac{2x+5}{3x-2}=\frac{3x^3+6x+12}{(x+2)(3x-2)}$$

[illegible]

5.3.6 Simplify 函数

[illegible]

```
R = simplify(S)
```

[illegible]

[illegible]

例 5.16 在 MATLAB 中使用 simplify 函数简化符号表达式。

step 1 在命令窗口输入以下内容：

```
>> syms a b positive
>> syms t x real
>> R1=simplify(sqrt(a^2+2*a*b+b^2));
>> R2=simplify(2^a*2^b);
>> R3=simplify(sec(t)^2-tan(t)^2);
>> [illegible]
>> R=[R1;R2;R3;R4];
```

step 3 在MATLAB命令窗口中输入"R"，按回车键，则[illegible]

```
[illegible]
[illegible]
[illegible]
2^(a+b)
[illegible]
[illegible]
```

从上述结果中可以看出，在MATLAB中使用simplify函数可以实现各种数学的化简变换，包括指数、对数和三角函数等。

5.3.7 Simple 函数

在MATLAB中，[illegible]函数[illegible]表达式转换成最简单的形式，其常见的调用格式如下：

- [illegible]函数，[illegible]表达式进行[illegible]化，[illegible]最简化形式，并返回其中最短的一个表达式。
- [illegible]，[illegible]使用的简化方法。r是符号表达式的结果，how则是使用的方法。

函数[illegible]，[illegible]simple函数[illegible]的简化方法。

表 5.1 simply 调用的简化方法

符号表达式	简化结果	调用的方法
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
cos(x)^2-sin(x)^2	cos(2*x)	combine
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	convert
(x+1)*x*(x-1)	x^3-x	collect
x^3+3*x^2+3*x+1	(x+1)^3	factor
[illegible]	[illegible]	[illegible]

[illegible]simple函数[illegible]

调用函数的使用环境

- 函数 radsimp 用来处理包含根式的表达式
- 函数 [illegible] 表达式的[illegible]
- 函数 convert 将一种形式转换成另一种形式。

例 5.17 在 MATLAB 中使用 simple 命令化简表达式 $\sqrt{\frac{1}{x^2}+\frac{1}{(x-1)^2}-2\times\frac{1}{x(x-1)}}$。

step 1 在命令窗口输入以下内容：

```
>> syms x positive
>> f=sqrt(1/x^2+1/(x-1)^2-2*1/(x*(x-1)));
```

step 2 为了方便查看 MATLAB 使用 simple 命令化简表达式的结果，[illegible]，以在命令窗口中输入"simple(f)"，然后按"Enter"键，得到的结果如下：

```
>> simple(f)
 simplify:
 1/x*abs(x-1)
 radsimp:
 [illegible]
 combine(trig):
 [illegible]
 factor:
 [illegible]
  exp: 1.
 [illegible]
  combine:
 [illegible]
  convert(exp):
 [illegible]
 convert(sincos):
 [illegible]
  convert(tan):
 [illegible]
  collect(x):
 [illegible]
 [illegible]
 (1/x^2+1/(x-1)^2-2/(x-1)/x)^(1/2)
 ans =
 [illegible]
```

说明 从上面的结果中可以看出，MATLAB 会调用不同的命令，[illegible]，然后比较所有的化简结果，选择其中最简化的结果，在本本中化简结果为 [illegible]

step 3 为了[illegible]，[illegible]，[illegible]，在命令窗口中输入下列表达式，得到的结果如下：

```
>> R=simple(f);
>> R
 R =
```

[illegible]

[illegible]，MATLAB [illegible]的中间结果。

根据基础数学知识，可以将表达式化简如下

$$\left(\frac{1}{x}+\frac{1}{(x-1)}\right)^2-2\times\frac{1}{x(x-1)}=\left(\frac{1}{x}+\frac{1}{x-1}\right)^2-\frac{1}{x}\times\frac{1}{x-1}=\frac{1}{x^2}+\frac{1}{(x-1)^2}+\frac{1}{x(x-1)}$$

[illegible]

例 5.18 在 MATLAB 中使用 [illegible] 命令简化表达式 $\cos(\theta)+(-\sin(\theta))^2$

Step 1 在命令窗口输入以下内容

```
>> syms theta
>> p=cos(theta)+sqrt(-sin(theta)^2);
>> R1=simple(p);
>> R2=simple(R1);
```

Step 2 在 MATLAB 命令窗口中输入 "R=[R1;R2]"，按下"Enter"键，[illegible]

```
>> R=[R1;R2]
 R =
[illegible]
[illegible]
```

[illegible]，[illegible]使用 [illegible]，[illegible]R1 和 R2 都是化简的步骤

$$\cos(\theta)+(-\sin(\theta))^2=\cos\theta+\sin\theta=e^{i\theta}$$

[illegible]

5.3.8 Pretty 函数

在 MATLAB 中，pretty 函数[illegible]，其[illegible]格式如下

- pretty(S)：将符号表达式[illegible]，使用默认的[illegible]
- pretty(S,n)：将符号表达式[illegible]，使用[illegible]

例 5.19 在 MATLAB 中使用 pretty 函数显示数值矩阵。

Step 1 在 MATLAB 命令窗口中输入[illegible]

```
>> A = sym(Pascal(3));
>> B=eig(A);
```

step 2 查看符号矩阵B，在命令窗口中输入下面的语句

```
>> B
B =
[illegible]
>> pretty(B)
```

[illegible]

在下面的解析代码中，Pretty 命令可以将符号表达式转换为 MATLAB 上的显示格式，而在下面的程序代码中分别显示出矩阵与数组的初始结构。

例5.20 在MATLAB中使用pretty()函数显示符号表达式。

step 1 在MATLAB的命令窗口中输入下面语句

```
>> syms x
P = x^2 - 1/(x^2+1)*x+sin(1/(1+x^2))
```

step 2 查看符号表达式的显示效果，在命令窗口中输入下面的语句

```
>> pretty(P)
```

[illegible]

从上述结果可以看出，pretty()函数会[illegible]以符号形式表示。

例5.21 在MATLAB中使用solve()函数[illegible]一元二次方程。

step 1 在MATLAB的命令窗口中输入下面的语句

```
>> syms a b c x
>> f=solve('a*x^2+b*x+c');
```

step 2 查看[illegible]，在命令窗口中输入下面的语句

```
>> pretty(f)
```

```
                    2          1/2]
        -b + (b  - 4 a c)     ]
1/2 ----------------------]
                  a           ]
```

```
[                2          1/2]
[      -b - (b   - 4 a c)     ]
[ 1/2   -----------------------]
[                a             ]
```

5.4 符号表达式的替换

在MATLAB中，符号计算的结果一般比较复杂，显得比数值计算烦琐，其中一个主要原因是有些子表达式多次出现在不同的地方。为了解决这个问题，MATLAB提供通过符号替换的方式来使表达式的输出形式简化，进而得到比较简单的表达式。

在MATLAB的符号工具箱中，提供两个函数subexpr和subs，实现符号表达式的替换，下面分小节详细介绍这两个命令。

5.4.1 Subexpr函数

在MATLAB中，subexpr函数的功能是将表达式中重复出现的字符串用变量替换，其常用的调用格式如下。

- [Y,SIGMA] = subexpr(X,SIGMA)　指定用变量SIGMA的值（该变量必须是符号对象）来替代符号表达式中重复出现的字符串。替换的结果由变量Y返回，被替换的字符串则由变量SIGMA代替。
- [Y,SIGMA] = subexpr(X,'SIGMA')　这种形式和上一种形式的区别在于，第二输入参数是字符或者字符串，它用来替换符号表达式中重复出现的字符串。

例5.22　在MATLAB中，使用相关命令来求解方程$ax^3+bx^2+cx+d=0$的通解表达式。

step 1 在MATLAB的命令窗口中输入下面的内容：

```
>> syms a b c d x
>> t = solve('a*x^3+b*x^2+c*x+d = 0');
```

step 2 在MATLAB的命令窗口中输入"R=subexpr(t)"，然后按"Enter"键，得到系统默认的化简结果如下：

```
>> R=subexpr(t)
 sigma =

36*c*b*a-108*d*a^2-8*b^3+12*3^(1/2)*(4*c^3*a-c^2*b^2-
18*c*b*a*d+27*d^2*a^2+4*d*b^3)^(1/2)*a

R =
1/6/a*sigma^(1/3)-2/3*(3*c*a-b^2)/a/sigma^(1/3)-1/3*b/a
 -1/12/a*sigma^(1/3)+1/3*(3*c*a-b^2)/a/sigma^(1/3)-1/3*b/a
+1/2*i*3^(1/2)*(1/6/a*sigma^(1/3)+2/3*(3*c*a-b^2)/a/sigma^(1/3))
 -1/12/a*sigma^(1/3)+1/3*(3*c*a-b^2)/a/sigma^(1/3)-1/3*b/a
-1/2*i*3^(1/2)*(1/6/a*sigma^(1/3)+2/3*(3*c*a b^2)/a/sigma^(1/3))
```

step 3 接着在MATLAB的命令窗口中输入下面的内容：

```
[r,s] = subexpr(t,'s');
```

step 6 然后，在 MATLAB 的命令窗口中依次输入 s、r，查看化简结果如下

```
s =

[illegible]
[illegible]

[illegible]
[illegible]
[illegible]
[illegible]
a*s^(1/3)+2/3*(3*c*a-b^2)/a/s^(1/3))
```

[illegible]

说明 在MATLAB中，[illegible] pretty [illegible] 在pretty命令中，[illegible] 达式，也不会被替换。

5.4.2 Subs 函数

在 MATLAB 中，subs函数[illegible] subexpr 命令，subs命令是一个通用的替换命令。subs 命令的常用调用格式如下。

- R = subs(S)：[illegible] 号变量的值，则返回值中该符号变量不被替换。
- R = subs(S, new)：[illegible] 变量的规则和函数 findsym 规则相同。
- R = subs(S,old,new)：[illegible] 串形式。

例 5.23 在 MATLAB 中，使用 subs 函数[illegible]

step 1 在 MATLAB 命令窗口中输入以下的代码

```
>> clear all
[illegible]
syms y
y = dsolve('Dy = -a*y');
```

step 2 [illegible]

```
>> [y,subs(y)]
ans =

  C1*exp(-a*t)
  [illegible]
```

step 3 在 MATLAB 的命令窗口中输入下面的内容

```
>> syms alpha theta t
>> P=subs(exp(theta*t),'theta',-magic(2));
```

step 4 查看上面的替换结果

```
>> P
 P =

    [illegible]
[ exp(-4*t), exp(-2*t)]
```

step 5 在 MATLAB 的命令窗口中输入下面的内容

```
>> syms theta alpha gamma delta
>> R=subs(cos(theta)+sin(alpha), [theta,alpha], [gamma,delta])
```

step 6 查看上面的替换结果

```
>> R
 R =

cos(gamma)+sin(delta)
```

[illegible]

5.5 符号函数的操作

在 MATLAB 中，在处理符号对象的时候，除了需要了解前面小节中介绍的符号表达式内容之外，还需要了解关于符号函数的相应操作。在 MATLAB 中的符号，因箱中，提供了关于符号函数的求导和复合函数，下面详细介绍这两类函数的应用。

5.5.1 finverse 函数

在 MATLAB 中，finverse 函数主要用于求解函数的反函数。反函数是高等数学的一个基本内容，其基本概念如下：对于函数 $f(x)$，在定义域 I 内，存在函数 $g(y)$，使得 $g(f(x))=x$，则函数 $g(y)$ 就被称为函数 $f(x)$ 的反函数。

finverse 命令的常见调用格式如下

- g = finverse(f)　对默认的自变量求反函数 g。
- g = finverse(f,v)　对指定的自变量为 v 的 f(v)，求反函数 g(v)。

例 5.24　在 MATLAB 中，分别求解不同类型函数的反函数。

step 1 [illegible]

```
>> syms t x
>> f1=finverse(log(t));
[illegible]
[illegible]
[illegible]
>> g=[f1;f2;f3;f4];
```

step 2 [illegible]

```
>> g

            exp(t)
        1/2*asin(t)
 1/2*t-1/2*log(x)
[illegible]
```

[illegible]

例 5.25　在 MATLAB 中，求解函数 $f(t)=t^2$ 的反函数。

step 1 在 MATLAB 的命令窗口中输入下列内容

```
>> syms t
>> f1=t^2;
[illegible]
```

step 2 [illegible]

```
Warning: finverse(t^2) is not unique.
> In sym.finverse at 43
 [illegible]
[illegible]
```

[illegible]，在本[illegible]结果是 $\sqrt{t}$，而没有显示另外一个反函数 $-\sqrt{t}$。

5.5.2 Compose 函数

在 MATLAB 中，compose 函数[illegible]复合函数，复合函数是数学分析中经常遇到的一[illegible]，其基本做[illegible]函数 f(t)= t^2，[illegible]复合函数为 f(g(x))。

在 MATLAB 中，compose 命令的常用调用格式如下

- [illegible] 对 $f(g)$ 和 $u=g(g)$ 下的复合函数 $fg=f(g(g))$。
- [illegible] 对函数 $f(x)$ 和 $v=g(y)$，求解复合函数 $fg=f(g(y))$。

例 5.26 在 MATLAB 中，实现复合函数。

Step 1 在 MATLAB 命令窗口中输入下列语句

```
>> syms x y z t u;
>> f = [illegible]
>> g = sin(y);
[illegible]
>> p = exp(-y/u);
>> fgy=compose(f,g);
fgt=compose(f,g,t);
[illegible]
[illegible]
[illegible]
hpx=compose(h,p,t,u,z);
>> comp= fgy;fgt;hgt;hgx;hpt;hpx]
```

Step 2 在 MATLAB 命令窗口中输入"comp"，然后按"Enter"键，查看使用复合函数创建的结果。

```
>> comp
comp =

[illegible]
1/(1+sin(t)^2)
        sin(z)^t
        [illegible]
    exp(-[illegible])^t
[illegible]exp(-y/[illegible])
```

从上面的结果来可以看出，对于相同的两个基础函数，如果使用不同的变量，复合函数的结果会有不同的结果，因此，在进行函数复合时，需要注意变量的选择。

在 MATLAB 中，compose(f,g)命令其实是 compose(f,g,cv,x)命令的默认形式，在这种情况下，x、y 都是由 findsym 函数自动对 f 和 g 函数进行默认确定的自变量。

5.6 符号微积分

在数学领域中，微积分一直是一个十分重要的内容，整个高等数学就是建立在微积分运算的基础上的，而，微积分也是微分方程体系的基础内容。在 MATLAB 中，提供一些常见的函数来支持这些微积分运算，所涉及的领域包括极限、微分、积分、级数等多个方面。

以符号数值计算为主，符号运算一般需要在内核中完成大量的计算，但是这并不意味着符号运算与数值计算完全分开，在有些情况下，符号计算处理比数值计算会更加便捷，下面将分别详细介绍符号运算在微积分中的应用。

diff 函数

当创建符号表达式后，就可以使用diff函数来对它们进行微分运算。在MATLAB中，diff命令的常用调用格式如下：

- diff(S,'v') 将符号‘v’当作变量，对符号表达式或者符号矩阵S求得微分；
- diff(S,n) 将符号表达式S中的默认变量进行n阶微分预算，其中默认的变量可以使用函数findsym来确定，参数n必须是正整数；
- diff(S,'v',n) 将符号‘v’当作变量，对符号表达式或者符号矩阵S求得n阶微分。

例5.27 在MATLAB中，使用diff函数来求解$f(x,y)=x^2+2xy-3y^2$的一阶和二阶偏导数值。

step 1 在MATLAB的命令窗口中输入下面的内容：

```
>> syms x y
>> f=x^2+2*x*y-3*y^2;
>>dfdx=diff(f,x);
>>dfdy=diff(f,y);
>>dfdxdy=diff(dfdx,y);
>>dfdydx=diff(dfdy,x);
```

step 2 在MATLAB的命令窗口中输入"[dfdx;dfdy;dfdxdy;dfdydx]"，然后按"Enter"键，可以得到下面的结果：

```
>> [ dfdx;dfdy;dfdxdy;dfdydx]

 ans =

 2*x+2*y
 2*x-6*y
       2
       2
```

从上面的程序代码中，通过使用简单的diff命令，就可以证明微积分中的一个结论：$\frac{\partial^2 f(x,y)}{\partial x\partial y}=\frac{\partial^2 f(x,y)}{\partial y\partial x}$。在本例中，$f(x,y)=x^2+2xy-3y^2$，因此根据上面程序的结果，两者满足$\frac{\partial^2 f(x,y)}{\partial x\partial y}-\frac{\partial^2 f(x,y)}{\partial y\partial x}=2$。

化简微分结果

例5.28 在MATLAB中，使用diff函数来求解常见的多项式$\ln(x+\sqrt{x^2\pm a^2})$的一阶导数数值，并使用MATLAB中相关的命令简化求解的结果。

step 1 在MATLAB的命令窗口中输入下面的内容：

```
>> syms x a
>> f1=diff(log(x+sqrt(x^2+a^2)));
>> f2=diff(log(x+sqrt(x^2-a^2)));
```

step 2 在MATLAB的命令窗口中依次输入f1、f2，查看求导结果如下：

```
[illegible]
>> f2
 f2 =
[illegible]
```

Step 3 [illegible]

```
[illegible];
>> D2=simple(f2);
>> D1
 D1 =
[illegible]
>> D2
 D2 =
[illegible]
```

根据[illegible]知识，多项式 $\ln(x+\sqrt{x^2\pm a})$ 的微分结果为

$$\frac{d}{dx}\ln(x+\sqrt{x^2\pm a})=\frac{1}{\sqrt{x^2\pm a}}$$

从上面的例子可以看出，diff函数的结果为一个表达式的微分，如果希望得到最简的微分结果，经常需要使用前面介绍的表达式化简命令。

5.6.3 求解矩阵微分

例5.29 在MATLAB中，求解多项式矩阵 $A=\begin{vmatrix} s\times\cos t & t\ln s \\ 2^s\times\sin(2t) & s^3/(2+t)\end{vmatrix}$ 的一阶导数微分数值 $\frac{dA}{dt}$、$\frac{d^2A}{ds^2}$、$\frac{d^2A}{dsdt}$。

Step 1 在MATLAB命令窗口中输入下面语句：

```
>> syms s t
>> [illegible];
>> df=diff(f);
>> dfds2=diff(f,s,2);
>> dfdsdt=diff(diff(f,t),s);
```

Step 2 在MATLAB命令窗口中依次输入上面语句，求解结果，查看本例结果如下：

```
>> df
 df =

[        -s*sin(t),        2*t*log(s)]
[  2*2^s*cos(2*t),         s^3/(2+t)]
```

```
>> dfds2
 dfds2 =

[                          0,                  -t^2/s^2]
[                          [illegible],                [illegible]]
>> dfdsdt
 dfdsdt =

[                   -sin(t),                     2*t/s]
[  2*2^s*log(2)*cos(2*t),              3*s^2/(2+t)]
```

[illegible]

5.6.4 向量微分 Jacobian 函数

[illegible] 向量函数 f 的 Jacobian 矩阵的定义如下：

[illegible] $f(v)=\begin{bmatrix} f_1(v) \\ \vdots \\ f_n(v) \end{bmatrix}$ [illegible]，其中函数 f [illegible]

$$f(v)=\begin{bmatrix} \frac{\partial f_1}{\partial v_1} & \cdots & \frac{\partial f_1}{\partial v_m} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial v_1} & \cdots & \frac{\partial f_n}{\partial v_m} \end{bmatrix}$$

若 f 是[illegible]，则表示[illegible]，[illegible]矩阵。

在 MATLAB 中，jacobian 的调用格式如下：

[illegible] jacobian(f,v) [illegible]，v 可[illegible]。

例 5.30 [illegible] A，[illegible] B_1 和 B_2，[illegible] jacobian 矩阵。

step 1 在 MATLAB 命令窗口中输入下面的语句：

```
>> syms s t
[illegible]
>> q1=f(:,1);
>> q2=f(:,2);
[illegible]
>> q1jacob=jacobian(q1,v);
>> q2jacob=jacobian(q2,v);
```

step 2 在 MATLAB 命令窗口中输入"q1jacob"和"q2jacob"，查看[illegible]矩阵：

```
>> q1jacob
```

```
g1jacob =

[                    cos(t),            -s*sin(t)]
[ 2^s*log(2)*sin(2*t),        2^s*2^s*cos(2*t)]
>> g2jacob
g2jacob =

[                    cos(t),            -s*sin(t)]
[ 2^s*log(2)*sin(2*t),        2^s*2^s*cos(2*t)]
```

根据例5.30，可以得知，$A=\begin{vmatrix} s\times\cos t & t^2\ln s \\ 2^s\times\sin(2t) & s^t\ln(2+t) \end{vmatrix}$，在本例中可以从中得到两个矩阵

$A_1=\begin{vmatrix} s\times\cos t & \\ 2^s\times\sin(2t) & \end{vmatrix}$ 和 $A_2=\begin{vmatrix} t^2\ln s & \\ s^t\ln(2+t) & \end{vmatrix}$，然后，可以求解上面两个矩阵的 jacobian 矩阵。

step 3 求解两个 jacobian 矩阵的行列式，得到的结果如下：

```
>> detJ1=simple(det(g1jacob));
>> detJ2=simple(det(g2jacob));
>> [detJ1;detJ2]

ans =

[                2^s*(2*cos(t)*cos(2*t)+s*sin(t)*log(2)*sin(2*t))]
[ 2^s*2^s*cos(t)*log(2*t)-s*log(s)*2^s*log(2+t)*t^t*s^s*sin(2*t)]
```

雅可比函数的第一个参数是列向量，而第二个参数则可能是行向量。同时，雅可比矩阵的行列式一般比较复杂，需要使用符号表达式的简化命令来进行计算。

5.6.5 符号极限

根据高等数学的基础知识，表达式的极限是微分的基础。极限的定义是当自变量的无穷小变化或者数值时，函数表达式的数值。本实验是中导微积分的基础理论，因此极限是整个微积分的基础。

在 MATLAB 中，提供函数 limit 来解表达式或者函数的极限，其常用的调用格式如下：

- limit(F,x,a)　求解当 x→a 时，符号表达式 F 的极限。
- limit(F,a)　符号表达式 F 采用默认自变量，该函数求解 F 的自变量趋近于 a 时的极限值。
- limit(F)　符号表达式 F 采用默认自变量，并且 x=0 为自变量的稳定值，该函数求解 F 的自变量趋近于 a 时的极限值。
- limit(F,x,a,'right')　该函数求解符号表达式 F 右极限，也就是自变量从右边趋近于 a 的函数极限值。
- limit(F,x,a,'left')　该函数求解表达式 F 的左极限，也就是自变量从左边趋近于 a 的函数极限值。

例 5.31　在 MATLAB 中，求解表达式 $\lim_{x\to 0}\frac{\tan x-\sin x}{x^3}$ 和 $\lim_{x\to 0}\frac{\tan x-x}{x^3}$ 的极限数值。

step 1 在 MATLAB 的命令窗口中输入下面的内容：

```
>> syms x
>> F1=limit((tan(x)-sin(x))/x^3);
>> F2=limit((tan(x)-x)/x^3);
```

Step 2 在命令窗口中输入"[F1,F2]"，查看极限表达式的数值如下

```
>> [F1,F2]

ans =

[ 1/2, 1/3]
```

根据例5.31中的相关说明，求解表达式字符串的极限可以使用两种方法，在MATLAB的示例中分别使用了这两种方法来求解各种极限数值。

上面的实例中使用了MATLAB的默认格式来求解极限数值，MATLAB会自动选择自变量x，则默认自变量x的趋近数值为0。

5.6.6 求解无限极限

例5.32 在MATLAB中，求解表达式"$\lim_{n\to\infty}\left(1+\frac{x}{n}\right)^n$"和"$\lim_{n\to\infty}\left(1-\frac{x}{n}\right)^n$"。

Step 1 在MATLAB的命令窗口中输入下面的内容

```
>> syms x
>> F1=limit( (1 + x/n)^n,n,inf );
>> F2=limit( (1 - x/n)^n,n,inf );
```

Step 2 在命令窗口中输入"[F1;F2]"，查看极限表达式的数值如下

```
>> [F1;F2]
ans =
  exp(x)
  exp(-x)
```

根据limit命令的格式，当极限的自变量不存在时，MATLAB中将会默认计算值，根据此结果证明MATLAB的相关命令正确。

5.6.7 求解左右极限

例5.33 在MATLAB中，求解$f(x)$的在$x=0$处的"左极限"和"右极限"数值，其中函数$f(x)$的"$x\neq0$"，$f(x)=\frac{x}{|x|}$；当$x=0$时，$f(x)=0$。

Step 1 在MATLAB的命令窗口中输入下面的内容

```
>> syms x
>> f1=limit(x/abs(x),x,0,'left');
   f2=limit(x/abs(x),x,0,'right');
>> f=limit(x/abs(x),x,0);
```

step 2 在命令窗口中输入"[f1,f2,f]"，查看极限表达式的数值如下

```
>> [f1,f2,f]
ans =

  -1
   1
 NaN
```

从上面的运算结果可以看出，左极限的数值为-1，右极限的数值为1，由于左右极限数值不等，所以得出结果极限不存在，在MATLAB中用NaN来表示。

step 3 绘制函数图像，由于上面表达式的极限值，所以绘制函数图像可以来帮助了解函数性态，相应的程序代码如下

```
>> x1=linspace(-1,0);
  y1=x1./abs(x1);
>> x2=linspace(0,1);
y2=x2/abs(x2);
plot(x1,y1,'r',x2,y2,'r')
axis([-1 1 -1.4 1.4])
text('x','y')
```

得到的图形如图5.1所示。

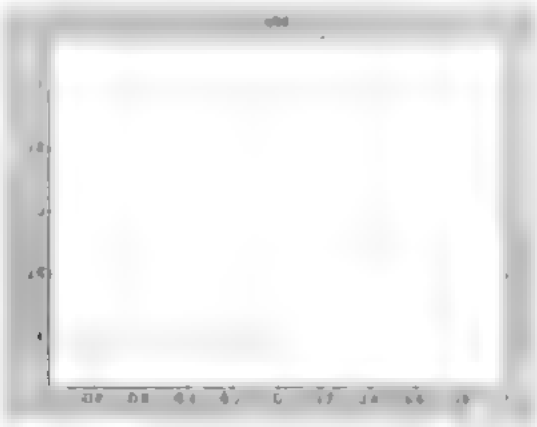


图5.1 绘制函数的图形

从上面的图形可以，函数图形是间断的，左极限为数值-1，右极限为数值1，因此函数在x=0的地方不存在极限数值。

上面所使用的绘图命令是函数的基本绘图命令，如果需要了解更多的绘图命令，可以参考本书中关于MATLAB绘图的章节。

5.6.8 符号积分

[illegible]

积分。int 命令可以直接调用 MAPLE，进行十分有效的求积。

[illegible]

式并显示警告信息。

在 MATLAB 中，int 命令的常用调用格式如下。

- R = int(S)　用默认的变量求符号表达式 S 的不定积分。
- [illegible]
- [illegible]
- [illegible]

[illegible]

例 5.34　[illegible]

step 1　[illegible]

```
[illegible]
[illegible]
>> F1=simple(I1);
[illegible]
>> F2=simple(I2);
>> F=[F1;F2];
```

step 2　[illegible]

```
>> F
[illegible]
```

根据上面的结果可以看出，积分结果满足下列的积分等式

$$\int \frac{1}{\sqrt{x^2 \pm a^2}} dx = \ln(x + \sqrt{x^2 \pm a^2}) + C$$

[illegible]

[illegible]

5.6.9 矩阵积分

例 5.35 在 MATLAB 中，求矩阵 $A=\begin{bmatrix}\cos t & t^2\\ 2^t & \ln(2+t)\end{bmatrix}$ 的积分结果。

step 1 在 MATLAB 的命令窗口中输入下面的内容：

```
>> syms t
>> f=[cos(t),t^2;2^t,log(2+t)];
>> I=int(f);
```

step 2 在命令窗口中输入"I"，然后按"Enter"键，查看定积分的计算结果：

```
>> I
 I =

[                sin(t),                1/3*t^3]
[         1/log(2)*2^t,  log(2+t)*(2+t)-2-t]
```

step 3 在命令窗口中输入"pretty(I)"，然后按"Enter"键，查看新的结果：

```
>> pretty(I)

                          [                          3          ]
                          [ sin(t)                1/3 t         ]
                          [                                     ]
                          [    t                                ]
                          [   2                                 ]
                          [ ------     log(2 + t) (2 + t) - 2 - t]
                          [ log(2)                              ]
```

从上面的计算结果中可以看出，当积分对象是符号矩阵的时候，int 命令会对矩阵中的元素依次进行积分，得出积分结果。

5.6.10 证明积分等式

例 5.36 在 MATLAB 中，使用 int 积分命令，证明下面的等式结果。

step 1 在 MATLAB 的命令窗口中输入下面的内容：

```
>> syms a positive
>> syms x;
>>f = exp(-a*x^2);
>>R=int(f,x,-inf,inf);
```

step 2 在命令窗口中输入"R"，然后按"Enter"键，查看定积分的计算结果：

```
>> R

R =
```

[illegible]

step 3 在MATLAB的命令窗口中输入下面的代码：

```
>> syms x
>> syms a real
>> f=exp(-a*x^2);
>> F = int(f, x, -inf, inf);
```

step 4 在命令窗口中输入“F”，然后按“Enter”键，查看定积分的计算结果：

```
>> F
 F =

PIECEWISE([ 1/a^(1/2)*pi^(1/2), signum(a) = 1],[ Inf, otherwise])
```

step 5 查看符号结果，在命令窗口中输入“pretty(F)”，然后按“Enter”键，查看简化后的定积分结果：

```
>> pretty(F)
                                  {  1/2
                                  { pi
                                  { -----          signum(a~) = 1
                                  {  1/2
                                  { a
                                  {
                                  {  Inf            otherwise
```

根据概率论中的[illegible]，$\frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{+\infty} e^{-\frac{(x-\mu)^2}{2\sigma^2}}dx=1$，[illegible]，得到 $\int_{-\infty}^{+\infty} e^{-ax^2}dx=\sqrt{\frac{\pi}{a}}$，[illegible]。

说明

在上面的程序代码中[illegible]syms a real[illegible]MATLAB[illegible]clear[illegible]MAPLE[illegible]“syms a unreal”[illegible]MAPLE[illegible]。

5.6.11 交互近似积分

除了上面提供的int命令之外，MATLAB还提供[illegible]rsums[illegible]函数在某个区间的[illegible]。在MATLAB中，rsums[illegible]rsums(f,a,b)，其中f是积分表达式，a和b分别为积分的上下限。

例5.37 在MATLAB下，使用rsums[illegible]函数 $f(x)=(x-1)^2+x+4x$ [illegible]结果。

step 1 在 MATLAB 的命令窗口中输入下面的内容

```
>> syms x;
>> f=exp(-1/(1+x^2))+4*x;
>> csums(f,-1,2)
```

step 2 在命令窗口输入上述语句后，按"Enter"键，MATLAB 会以动画方式显示积分的交互过程，如图 5.2 所示。

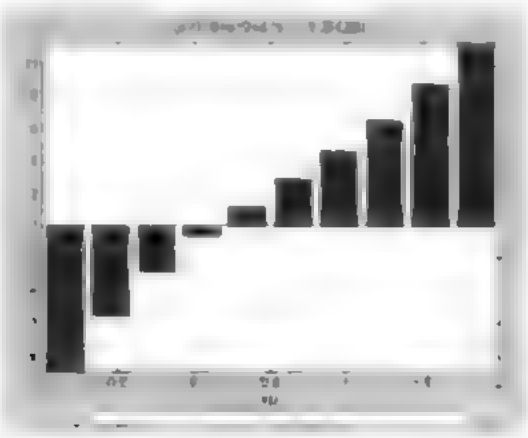


图 5.2 交互近似积分界面

在默认的情况下，程序会在积分区间内设置一个"滑动键"，滑动键用来设置函数曲线下方的矩形个数，默认情况下，矩形的个数为 1，当"滑动键"向右滑动时，矩形数会增加。最大的矩形个数为 128。

step 3 调整积分矩形个数，可在调整"滑动键"，使其位置为 128，观察近似积分的数值，如图 5.3 所示。

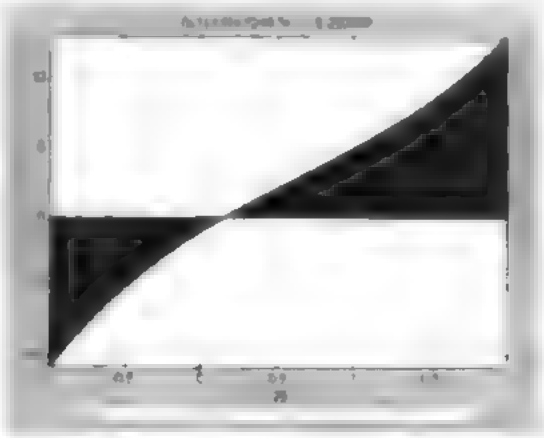


图 5.3 调整积分精度

step 4 在命令窗口中输入"int(f,-1,2)"，计算函数 f 在区间[-1,2]上的积分数值，结果如下

```
>> int(f,-1,2)
ans =
```

```
21/4
```

从上面的结果中可以看出，精确值为21/4=5.25，而上面的计算结果的数值为5.0000，精度不够，因此要求

5.6.12 符号级数求和

在数学分析中，级数求和是一个重要的内容，MATLAB提供symsum命令来对解析表达式进行求和。symsum命令的调用格式如下

- s = symsum(f,a,b)：将符号表达式f中的默认变量从a变到b时的有限和
- s = symsum(f,v,a,b)：将符号表达式f中的变量v从a变到b时的有限和

例5.38 在MATLAB中，使用symsum命令来解决一些级数求和

step 1 在MATLAB的命令窗口中输入下面的内容

```
>> syms x k
>>s1=symsum(1/k^2,1,inf);
>> s2=symsum(k^2);
>> s3=symsum(x^k/sym('k!'), k, 0,inf) ;
```

step 2 在命令窗口中输入"S=[s1;s2;s3]"，然后按"Enter"键，查看求解的结果如下

```
>> S=[ s1;s2;s3]
S =

                    2*exp(x)
 1/6*pi^2 + 1/2*k^2 + 1/2*k + 1/3*k^3
                    exp(x)
```

上面的计算结果分别对应的级数运算如下

$$\sum_{k=1}^{\infty}\frac{1}{k^2}=\frac{\pi^2}{6}$$

$$\sum_{k=1}^{n}k^2=\frac{2k^3+3k^2+k}{6}=\frac{k(k+1)(2k+1)}{6}$$

$$\sum_{k=1}^{\infty}\frac{x^k}{k!}=e^x$$

关于上面各种级数求解的结果，可以查阅相关的数学书籍

在上面的表达式中，将sym命令将"k!"转换为符号变量，在后面的计算中可以，MATLAB会将"k!"作为阶乘。由于MATLAB中k未定义，所以在计算不会出现错误，MATLAB会提示错误信息

5.7 符号积分变换

在数学分析中，通过数学变换将复杂的计算转换为简单的计算是一个重要的手段。其中，积分变换是数学变换中的一个重要内容。所谓积分变换，就是通过积分计算，把一个函数 A 变换成另一个函数 B，而函数 B 一般是定在参量 ω 的积分 $\int_a^b f(t)K(t,\omega)dt$。这一变换的目的就是，将函数类 A 中的函数 $f(t)$ 通过积分运算变成另外一个函数类 B 中的函数 $f(\omega)$。积分变换计算，其中的 $K(t,\omega)$ 称为积分变换的核，而 $f(t)$ 叫做原函数，$f(\omega)$ 叫做像函数。

积分变换的方法在自然科学和工程技术中有着广泛的应用。这一节，我们来讲解 MATLAB 中，将介绍一个重要的积分变换：Fourier 变换、Laplace 变换和 Z 变换。

5.7.1 Fourier 变换命令

在时域中的 $f(t)$ 与在频域中的 Fourier 变换 $F(w)$ 之间存在着如下的关系

$$F(w)=\int_{-\infty}^{\infty}f(t)e^{-jwt}dt$$

$$f(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty}F(w)e^{jwt}dw$$

根据上面的积分定义，可以使用前面介绍的命令直接求解，不过，MATLAB 中提供了 Fourier 变换和逆 Fourier 变换的命令 fourier 和 ifourier 命令。在本节中，将详细介绍 MATLAB 中的这两个命令，以及积分定义的方法。感兴趣的读者请自行尝试。

在 MATLAB 中，Fourier 和 ifourier 命令的调用格式如下

- Fw=fourier(f,t,w)：表示时域上函数 f 的 Fourier 变换 Fw，其中 f 是以 t 为自变量的函数，Fw 是以圆频率 w 为自变量的频域函数。
- f=ifourier(Fw,w,t)：表示频域上函数 Fw 的 Fourier 逆变换 f，其中 f 是以 t 为自变量的时域函数，Fw 是以圆频率 w 为自变量的频域函数。

5.7.2 Fourier 变换实例

例 5.39 在 MATLAB 中，分别使用 fourier 命令和积分定义，求解表达式 $f=e^{-x^2}$ 和 $f=e^{-|x|}$ 的 Fourier 变换。

Step 1 在 MATLAB 的命令窗口输入下面的语句

```
>> syms x w t
>>f1=exp(-x^2);
>>g1=f1*exp(-i*w*x);
>>fy1=fourier(f1);
>>fd1=int(g1,x,-inf,inf);
>> f2=exp(-abs(w));
>>g2=f2*exp(-i*w*t);
>> fy2=fourier(f2);
>>fd2=int(g2,w,-inf,inf);
```

在上面的程序代码中，fy1 和 fy2 是使用 Fourier 变换命令求解的计算结果，而 fd1 和 fd2 是根据积分定义，计算得到的结果。读者可以分别将 fd1 和 fd2 与 fy1 和 fy2 进行比较，看看两种方法的计算结果是否一致。

Step 7 在 MATLAB 的命令窗口中输入计算结果的数组

```
>> [ fy];simple(fd1)]
ans =
  pi^(1/2)*exp(-1/4*w^2)
  [ (2*pi)^(1/2)*exp(-1/4*w^2)
>> fy2=simple(fd1)
fy2 =
[ (2*pi)^(1/2)*exp(-1/4*w^2)
  0]
```

上面的程序代码中计算的积分表达式依次如下

$f(t)=e^{-t^2}$，$F(w)=\int_{-\infty}^{\infty} f(t)e^{-jwt}dt$，根据积分运算，得到的结果是 $\sqrt{\pi}e^{-\frac{w^2}{4}}$。

$f(w)=e^{-w^2}$，$F(t)=\int_{-\infty}^{\infty} f(w)e^{-jwt}dw$，根据积分运算，得到的结果是 $\frac{2}{1+t^2}$。

说明

[illegible]

例 5.40 使用 fourier 函数计算表达式 $\int_{-\tau/2}^{\tau/2} Ae^{-jwt}dt = A\tau g\frac{\sin\frac{w\tau}{2}}{\frac{w\tau}{2}}$

Step 1 在 MATLAB 的命令窗口输入下面的内容

```
>> syms A t w
>> syms tao positive
>> yt=sym('Heaviside(t+tao/2)-Heaviside(t-tao/2)');
>> Yw=fourier(A*yt,t,w);
```

Step 2 在 MATLAB 的命令窗口中输入 "Yw"，查看 fourier 的结果如下

```
>> Yw
Yw =

[illegible]
```

[illegible] 了能够使用 fourier 变换求解积分表达式的数值，[illegible] 函数，该函数的数学含义是单位阶跃函数，当自变量 t>=a 时，函数的数值为 1，[illegible] 函数的数值为 0。

根据上面的程序，在命令窗口中输入的 'Heaviside(t+tao/2)-Heaviside(t-tao/2)' 的数值在自变量 $t\in\left[-\frac{\tau}{2},\frac{\tau}{2}\right]$ 为 1，在其他自变量取值时为 0。这恰恰与要相求的积分表达式的积分范围一致。

根据上面的程序代码，fourier 函数求解的结果为 $\frac{2A}{w}g\sin\frac{w\tau}{2} = A\tau g\frac{\sin\frac{w\tau}{2}}{\frac{w\tau}{2}}$，因此上面的程序代码就等于证明了 $\int_{-\tau/2}^{\tau/2} Ae^{-jwt}dt = A\tau g\frac{\sin\frac{w\tau}{2}}{\frac{w\tau}{2}}$。

在MATLAB的符号数学工具箱中，增加了dirac和heaviside两个函数，它们的功能分别是单位脉冲函数和阶跃函数。在以前的版本中，MATLAB只是调用了MAPLE函数库中的heaviside函数来定义，也就是说在MATLAB中使用这个函数时，必须加入MAPLE函数库中调用。

5.7.3 Laplace 变换命令

在数学分析中，Laplace 变换和反变换的定义如下

$$F(s)=\int_0^{\infty} f(t)e^{-st}dt$$

$$f(t)=\frac{1}{2\pi j}\int_{c-j\infty}^{c+j\infty} F(s)e^{st}ds$$

由于拉氏变换中涉及到积分定义的，所以，MATLAB 的命令直接求解Laplace变换。不过，MATLAB中提供专门的Laplace变换程序：laplace和ilaplace命令。在本小节中，将详细介绍MATLAB中的这些命令，对于积分定义的方法，感兴趣的读者请自行尝试。

在 MATLAB 中，laplace 和 ilaplace 命令的调用格式如下：

- **Fs=laplace(ft,t,s)**：求时域上函数ft的Laplace变换Fs，其中ft是以t为自变量的时域函数，Fs是以复频率s为自变量的频域函数
- **ft=ilaplace(Fs,s,t)**：求频域上函数Fs的Laplace反变换ft，其中ft是以t为自变量的时域函数，Fs是以复频率s为自变量的频域函数。

5.7.4 Laplace 变换实例

例 5.41 证明Laplace变换的时移性质 $L\{f(t-t_0)u(t-t_0)\}=e^{-st_0}L\{f(t)\}$。

其中，f是任意一个函数，u是单位阶跃函数，t_0是一个非负的实数，L代表的是Laplace变换。

Step 1 在 MATLAB 命令窗口输入下面的内容

```
>> syms t s
>> syms t0 positive
>> ft=heaviside(t-t0)*sym('f(t-t0)');
>> FS=laplace(ft,t,s);
>> FS_t=ilaplace(FS,s,t);
```

Step 2 在 MATLAB 命令窗口中，查看上面符号分析变化的结果

```
>> ft
 ft =
 heaviside(t-t0)*f(t-t0)
>> FS
 FS =
 exp(-s*t0)*laplace(f(t),t,s)
>> FS_t
 FS_t =
 heaviside(t-t0)*f(t-t0)
```

从上面的结果可以看出，$ft=f(t-t_0)u(t-t_0)$，FS就是ft函数对应的Laplace变换的结果，

$\mathcal{L}\{f(t-t_0)u(t-t_0)\}=e^{-st_0}\mathcal{L}\{f(t)\}$ [illegible] 的[illegible]，[illegible]，[illegible]定 $f(t)=f(t-t_0)u(t-t_0)$。

[illegible]

5.7.5 Z 变换命令

[illegible]

$$F(z)=\sum_{n=0}^{\infty} f(n)z^{-n}$$

$$f(n)=\mathcal{Z}^{-1}\{F(z)\}$$

[illegible] 在 MATLAB 中，[illegible]，其表达式为 $f(n)=\frac{1}{2\pi j}\oint_C F(z)z^{n-1}dz$ [illegible]

- f1=ztrans(fun,n,z) [illegible] 的 Z 变换 f1，[illegible]，f1 是以复频率 z 为自变量的频域函数。
- f2 = [illegible] 的 Z 变换 f2，[illegible]，f2 是以复频率 z 为自变量的频域函数。

5.7.6 Z 变换实例

例 5.42 在 MATLAB 中，对下面的函数实现 Z 变换。

step 1 在 MATLAB 的命令窗口输入下面的命令

```
>> syms n z w a
>> f1 = [illegible] 4;
>> f2 = [illegible]
>>f3=sin(a*n);
>> z1=ztrans(f1);
>> z2=ztrans(f2);
>> z3=ztrans(f3,n,
```

step 2 在 MATLAB 的命令窗口输入 [illegible]，函数 Z 变换后的表达式如下

```
>> [simple(z1);simple(z2);simple(z3)]
ans =
  [illegible]
                        -w/(-w+a)
  [illegible]
```

例 5.43 在 MATLAB 中，对下面的函数实现 Z 的反变换。

step 1 在 MATLAB 的命令窗口输入下面的命令

```
>> syms z n k a
>> f1 = 2*z/(z-2)^2;
>> f2= n*(n+1)/(n^2+2*n+1);
>> f3= z/(z-a);
>> z1=iztrans(f1); z2=iztrans(f2);
>> z3=iztrans(f3);
```

step 2 在MATLAB的命令窗口输入"[z1;z2;z3]"，查看Z反变换后的表达式如下.

```
>> [ z1;z2;z3]
 ans =
  2^n*n
 (-1)^k
    a^n
```

符号矩阵的计算

矩阵计算 直都是MATLAB计算的核心内容。关于数值矩阵运算的内容在前面章节中已经介绍过，这里就不重复介绍了。而在MATLAB的符号工具箱中，符号矩阵的运算规则和数值矩阵的运算规则大致相同，没有添加新的运算规则，这就给用户使用符号矩阵带来了极大的方便。本节将主要介绍符号矩阵的一些简单应用。

线性代数运算

符号对象的线性代数运算和双精度的线性代数运算一样，读者可以查阅数值计算的相应内容来了解有关线性代数运算的一些规则和注意事项。在本小节中，将利用符号对象来进行相应的线性代数计算。

例 5.44 在MATLAB中使用符号对象进行基础的矩阵运算。

step 1 在MATLAB的命令窗口输入下面的内容：

```
>> syms t
>> [ I,J] =meshgrid(1:5);
>> H=1./(I+J-t);
```

在上面的程序代码中，引入了符号对象t，同时产生包含变量t的Hilbert矩阵H，这就相当于在Hilbert矩阵中引入了参数t。

step 2 查看创建的Hilbert矩阵H，如下：

```
>> H
 H =
[   1/(2-t),   1/(3-t),   1/(4-t),   1/(5-t),   1/(6-t)]
[   1/(3-t),   1/(4-t),   1/(5-t),   1/(6-t),   1/(7-t)]
[   1/(4-t),   1/(5-t),   1/(6-t),   1/(7-t),   1/(8-t)]
[   1/(5-t),   1/(6-t),   1/(7-t),   1/(8-t),   1/(9-t)]
[   1/(6-t),   1/(7-t),   1/(8-t),   1/(9-t),  1/(10-t)]
```

step 2 将符号对象t设置为参数1，MATLAB会自动计算出原始的Hilbert矩阵，使用的命令为前面

章节介绍过的Subs，结果如下

```
>> Hsb=Subs(H,t,1)
Hsb =
    1.0000    0.5000    0.3333    0.2500    0.2000
    0.5000    0.3333    0.2500    0.2000    0.1667
    0.3333    0.2500    0.2000    0.1667    0.1429
    0.2500    0.2000    0.1667    0.1429    0.1250
    0.2000    0.1667    0.1429    0.1250    0.1111
```

[illegible]

step 4 计算符号矩阵H的行列式的倒数

```
>> d=1/det(H)
d =
    -1[illegible]*(-2+t)*(-4+t)^3*(-6+t)^5*(-8+t)^2*(-10+t)*(-9+t)^2*(-7+t)
^4*(-5+t)^4*(-3+t)^2
```

由于H是关于t的符号矩阵，与通过det函数计算该符号矩阵的行列式，该行列式也是关于t的符号表达式，而且对应的倒数d也自然是关于t的符号表达式。

step 5 展开上述的符号表达式d，得到的结果如下

```
>> f=expand(d)
f =
[illegible]
-1078920141906600*t^3+742618453752000*t^2+25/13824*t^24-1/82944*t^25
[illegible]
[illegible]
414[illegible]
```

step 6 由于上面符号表达式d太复杂，所以对其进行简化，结果如下

```
>> pretty(f)
   10640296363350955  8                          2885896606895  16
  ----------------- t  - 323874210240000 t + ------------- t
          96                                        13824
           1588946776255  17   8194259295156385  13                    4
         - -------------- t  - ---------------- t   + 1115685328012530 t
               82944                  82944
           1268467075  18   2405139875  19   15940015  21   21896665  20
         + ---------- t   - --------- t   - -------- t   + -------- t
               864              2592            82944            4608
           40825  22   5375  23                    3                    2
         + ----- t   - ----- t   - 1078920141906600 t  + 742618453752000 t
```

```
  6912          41472
   25     24              25    19701982062369302 5   9
+ ----- t    - 178294 4 t    - ------------------ t
  13824                                 5184
  37909434298793825  10    55608098267105175  11
+ ------------------ t    - ------------------ t    + 67212633000000
        3456                              20736
                                5   12958201048605475  6   7707965723450845  12
- 1748754621252377/2 t   + ------------- t   + ----------- t
                                          24                          13824
  38821472549340925  7    34372691161375  14    79493630114675  15
- ------------- t   + --------- t    - --------- t --------- t
        144                          2304                          41472
```

> [illegible]

Step 1 计算H矩阵的逆矩阵X，保留字符变量t。

```
>> X=inv(H)
 X =
[                          -1/576*(-4+t)^2*(-6+t)^2*(-5+t)^2*(-2+t)*(-3+t)^2,
1/144*(-3+t)*(-6+t)^2*(-2+t)*(-5+t)^2*(-4+t)^2*(-2+t),
-1/36*(-3+t)*(-5+t)^2*(-9+t)*(-7+t)*(-8+t)^2*(-4+t)*(-2+t),
1/144*(-3+t)*(-5+t)*(-7+t)*(-8+t)*(-9+t)*(-4+t)^2*(-2+t)^2, [illegible]
(-3+t)*(-5+t)*(-7+t)*(-44+t)*(-84+t)*(-4+t)*(-1+t)*(-2+t)*[illegible],
[                      1/144*(-3+t)*(-6+t)^2*(-7+t)*(-5+t)^2*(-4+t)^2*(-2+t),
-1/36*(-6+t)^2*(-7+t)^2*(-4+t)*(-5+t)^2*(-3+t)^2,
1/24*(-5+t)*(-8+t)*(-7+t)*(-6+t)^2*(-4+t)^2*(-3+t),
-1/36*(-3+t)*(-6+t)*(-3+t)*(-7+t)^2*(-8+t)*(-5+t)*(-4+t)*[illegible],
1/144*(-5+t)*(-7+t)*(-4+t)*(-8+t)*(-6+t)^2*(-3+t)*[illegible]
[                 -1/9*(-3+t)*(-5+t)^2*(-8+t)*(-7+t)^2*(-4+t)^2*(-1+t)*(-2+t),
1/24*(-5+t)*(-8+t)*(-7+t)^2*(-6+t)^2*(-4+t)^2*(-3+t),
-1/16*(-4+t)^2*(-8+t)^2*(-6+t)*(-7+t)^2*(-5+t)^2,
1/24*(-4+t)*(-7+t)*(-8+t)^2*(-9+t)*(-5+t)^2*(-6+t)^2,
-1/36*(-4+t)*(-7+t)^2*(-9+t)*(-8+t)*(-5+t)^2*[illegible]
[          1/144*(-3+t)*(-5+t)*(-4+t)*(-8+t)*(-6+t)*(-7+t)^2*(-4+t)*(-2+t),
-1/36*(-3+t)*(-6+t)*(-9+t)*(-7+t)^2*(-8+t)*(-5+t)^2*(-4+t),
1/24*(-4+t)*(-7+t)*(-8+t)^2*(-9+t)*(-5+t)^2*(-6+t)^2,
-1/36*(-6+t)^2*(-8+t)*(-5+t)^2*(-7+t)^2*(-9+t)^2,
1/144*(-5+t)*(-7+t)^2*(-2+t)*(-4+t)*(-8+t)*(-6+t)*[illegible]
  -1/576*(-3+t)*(-7+t)*(-2+t)*(-4+t)*(-8+t)*(-6+t)*(-5+t)*(-1+t)*(-2+t),
1/144*(-5+t)*(-7+t)*(-4+t)*(-8+t)*(-6+t)^2*(-3+t)*[illegible],
-1/36*(-4+t)*(-7+t)^2*(-2+t)*(-5+t)*(-9+t)*(-8+t)^2*(-6+t)*(-3+t),
1/144*(-5+t)*(-7+t)^2*(-2+t)*(-8+t)*(-6+t)*(-3+t)*[illegible],
-1/36*(-8+t)^2*(-7+t)^2*(-4+t)*(-6+t)*(-1+t)*(-2+t)]
```

Step 2 计算Hilbert矩阵的逆矩阵，保留字符变量t。

```
>> subs(X,t,1)
ans =
   1.0e+005 *
    0.0003   -0.0030    0.0105   -0.0140    0.0063
   -0.0030    0.0480   -0.1890    0.2688   -0.1260
    0.0105   -0.1890    0.7938   -1.1760    0.5670
```

```
   -0.0140    0.2688   -1.1760    1.7920   -0.8820
    0.0063   -0.1260    0.5670   -0.8820    0.4410
```

在上面的程序代码中，将上述步骤中未知的x矩阵中参数t设置为1，就可以获得[illegible]矩阵的逆矩阵。

在上面的步骤中使用了多种数值矩阵的命令，例如[illegible]、inv等，与数值计算中的结果相比，这些符号命令都可以通用于符号表达式中。

5.8.2 特征值运算

特征值运算，是矩阵运算的重要内容。在符号矩阵中，同样提供数值矩阵常用的求解特征值的命令eig。同时，与之相对应的任意精度计算的命令是eig(vpa(A))。

例5.45 在MATLAB中，使用符号对象来进行矩阵运算。

step 1 在MATLAB的命令窗口输入下面的内容

```
>> A=sym(gallery(5));
>> B=A^5;
```

在上面的程序代码中使用代码sym(gallery(5))创建一个5阶的符号矩阵A，然后对该符号矩阵A进行乘幂计算得到矩阵B。

step 2 查看乘幂的结果矩阵B

```
>> B
 B =
[ 0,  0,  0,  0,  0]
[ 0,  0,  0,  0,  0]
[ 0,  0,  0,  0,  0]
[ 0,  0,  0,  0,  0]
[ 0,  0,  0,  0,  0]
```

可以看出，A的5次乘幂是零矩阵，根据线性代数的基础知识，A是零幂矩阵，其对应的所有特征值都应该是0。

step 3 求解符号矩阵A的特征多项式

```
>> p=poly(A,'lambda')
 p =
 lambda^5
```

在上面的程序代码中，使用poly命令来求解符号矩阵A对应的特征多项式，由于矩阵A的特征值全为0，因此其特征多项式就是一个5阶的幂表达式。

step 4 求解符号矩阵A的特征值

```
>> lambda=eig(A)
 lambda =
 0
 0
 0
```

```
 0
 0
```

从上面的程序结果中可以看出，通过符号矩阵的eig命令，可以直接求解数值矩阵的特征值，得到的结果为0。

step 5 求解任意精度下符号矩阵A的特征值，得到的结果如下：

```
>> lambda=eig(vpa(A))
 lambda =
  -.22428967199963540318045946788 22e-5+.16295641216458292389044088574558e-5*i
 -.22428967199963540318045946788 22e-5-.16295641216458292389044088574558e-5*i
   .85671437348734831659168924628 7e-6+.26366776411426003145716139438407e-5*i
   .85671437348734831659168924628 7e-6-.26366776411426003145716139438407e-5*i
                                                    .27723646930180114304258803230 80e-5
```

step 6 求解符号矩阵A的约当(Jordan)标准型，得到的结果如下：

```
>> J=jordan(A)
 J =
[  0,  1,  0,  0,  0]
[  0,  0,  1,  0,  0]
[  0,  0,  0,  1,  0]
[  0,  0,  0,  0,  1]
[  0,  0,  0,  0,  0]
```

step 5 求解符号矩阵A的矩阵指数，得到的结果如下：

```
>> E=expm(t*A)
 E =
 [                                1-9*t+11/2*t^2-2/3*t^3,
4/3*t^3-9*t^2+11*t,                                  -10/3*t^3+39/2*t^2-21*t,
32/3*t^3-58*t^2+63*t,                                -85/2*t^3+232*t^2-252*t]
[                        70*t-115*t^2+81/2*t^3-7/2*t^4,
1+7*t^4-67*t^3+301/2*t^2-69*t,          -35/2*t^4+293/2*t^3-299*t^2+141*t,
56*t^4-438*t^3+1799/2*t^2-421*t,        -1785/8*t^4+3503/2*t^3-3597*t^2+1684*t]
[                  -575*t+1717/2*t^2-285*t^3+71/3*t^4,
-142/3*t^4+1426/3*t^3-1146*t^2+575*t,       1+355/3*t^4-3139/3*t^3+4585/2*t^2-1149*t,
-1136/3*t^4+3140*t^3-6875*t^2+3451*t,       6035/4*t^4-75323/6*t^3+27496*t^2-13801*t]
[              3891*t-5837*t^2+11675/6*t^3-973/6*t^4,
973/3*t^4-3243*t^3+15565/2*t^2-3891*t,      -4865/6*t^4+14269/2*t^3-15565*t^2+7782*t,
1+7784/3*t^4-64210/3*t^3+93391/2*t^2-23345*t,
-82705/8*t^4+513437/6*t^3-373503/2*t^2+93365*t]
[                  1024*t-1536*t^2+512*t^3-128/3*t^4,
256/3*t^4-2560/3*t^3+2048*t^2-1024*t, -640/3*t^4+5632/3*t^3-4096*t^2+2048*t,
2048/3*t^4-5632*t^3+12288*t^2-6144*t,       1-2720*t^4+67552/3*t^3-49144*t^2+24572*t]
```

step 8 求解符号矩阵A的矩阵元素指数，得到的结果如下：

```
>> X=exp(t*A)
 X =
[     exp(-9*t),     exp(11*t),    exp(-21*t),     exp(63*t),   exp(-252*t)]
[     exp(70*t),    exp(-69*t),    exp(141*t),   exp(-421*t),   exp(1684*t)]
[   exp(-575*t),    exp(575*t),  exp(-1149*t),   exp(3451*t), exp(-13801*t)]
[   exp(3891*t),  exp(-3891*t),   exp(7782*t), exp(-23345*t),  exp(93365*t)]
```

exp(1/4*t), exp(1/24*t), exp(1/48*t), exp(1/144*t), exp(1/45*2*t)

[illegible]

5.9 符号代数方程的求解

[illegible]

5.9.1 solve 命令

[illegible] MATLAB 提供的一个专用命令 solve，[illegible]。[illegible]

在 MATLAB 中，solve 命令的调用格式如下。

- [illegible]
- [illegible]
- [illegible]
- [illegible]

[illegible]

5.9.2 求解非线性方程组

例 5.46 求解非线性方程组 $\begin{cases} x^2 - xy + y = 3 \\ x^2 - 4x + 3 = 0 \end{cases}$ 的数值解。

Step 1 在 MATLAB 的命令窗口中输入下面的内容

```
>> syms x y
>> [x,y] = solve('x^2 + x*y + y = 3','x^2 - 4*x + 3 = 0');
```

Step 2 在命令窗口中输入结果语句，得到运行结果如下

```
>> solution=[x,y]
solution =
[     1,     1]
[     [illegible]
```

从上面的求解结果可以看出，MATLAB会对所输入方程组的参数进行[illegible]的变量，为会对输出x和y的结果。因此，建议为[illegible]

5.9.3 求解含参数方程组

例5.47 求解含有参数a的非线性方程组$\begin{cases} au^2+v^2=0 \\ u-v=1 \end{cases}$的解。

Step 1 在MATLAB的命令窗口中输入下面的内容

```
>> syms u v a
>> [u,v]=solve('a*u^2+v^2=0','u-v=1');
>> solution=simple([u,v]);
```

Step 2 在命令窗口中输入结果矩阵，得到的结果如下

```
>> solution
 solution =
[   ((-a)^(1/2)+1)/(a+1), -(a-(-a)^(1/2))/(a+1)]
[  -((-a)^(1/2)-1)/(a+1), -(a+(-a)^(1/2))/(a+1)]
```

5.9.4 求解超越方程组

例5.48 求解超越方程组$\begin{cases} \sin(x+y)-e^x y=0 \\ x^2-y=2 \end{cases}$的解。

Step 1 在MATLAB的命令窗口中输入下面的内容

```
>> syms x y
>> S=solve('sin(x*y)-exp(x)*y=0','x^2-y=2');
```

Step 2 在命令窗口中输入结果矩阵，得到的结果如下

```
>> S
S =

    x: [2x1 sym]
    y: [2x1 sym]
```

上面的程序结果中，并没有显示方程组的解，只是显示方程结果的属性和维度。在本例中，变量x和y都是符号变量，维度都是2x1。

Step 3 查看各个变量的具体数值，得到的结果如下

```
>> S.x
 ans =
  -.33129879499763797066864098166363
  -.66870120500236202933135901833637
>> S.y
 ans =
  -1.8952411[illegible]
  -1.5[illegible]
```

[illegible]

5.10 符号微分方程的求解

[illegible]

5.10.1 dsolve 命令

[illegible]

- r = dsolve('eq1,eq2,...', 'cond1,cond2,...', V)
- [illegible]

[illegible]

[illegible]

5.10.2 求解常微分方程

例 5.49 [illegible] $\frac{dx}{dt}$ [illegible] 通解，然后求解在初值条件 x(0)=1 的数值解。

Step 1 [illegible]

```
>> syms x a
>> S1=dsolve('Dx=-a*x');
[illegible]
[illegible]
```

Step 2 在命令窗口中输入 "solution"，查看求解的结果。

```
>> solution
solution =
  C1*exp(-a*t)
    exp(-a*t);
```

在上面的程序代码中，S1是微分方程的通解，S2(t)是初值条件下的特解。在默认情况下，MATLAB会依次使用C1、C2作为通解的常量符号。

5.10.3 求解二阶常微分方程

例5.50 求二阶常微分方程 $\begin{cases}\dfrac{d^2y}{dx^2}=\cos(2x)-y\\ y(0)=1,y'(0)=0\end{cases}$ 的通解。

step 1 在MATLAB中的命令窗口中输入下面的命令

```
>> syms x y
S=dsolve('D2y=cos(2*x)-y','y(0)=1','Dy(0)=0','x');
```

step 2 在命令窗口中输入"S"，查看求解的结果

```
>> S
 S =
4/3*cos(x)^2-1/3*cos(2*x)
```

在上面的程序中，求解的方程为y(x)，而不是方程y(t)，如果在命令中没有特别指定方程自变量x，得到的结果将是关于自变量t的表达式，如下：

```
>> S1=dsolve('D2y=cos(2*x)-y','y(0)=1','Dy(0)=0');
>> S1
 S1 =
cos(t)*(1-cos(2*x))+cos(2*x)
```

在求微分方程时，用大写字母D表示一次微分，用D2、D3等表示二次、三次微分，而求解函数，再次提醒读者，MATLAB是区分大小写的，不要用小写字母，使用小写字母时会出现错误信息。

5.10.4 求解常微分方程组

例5.51 求常微分方程组 $\begin{cases}\dfrac{df}{dt}=3f(t)+4g(t)\\ \dfrac{dg}{dt}=-4f(t)+3g(t)\\ f(0)=0,g(0)=1\end{cases}$ 的通解。

step 1 在MATLAB中的命令窗口中输入下面的命令

```
>> syms f g
>> [f,g]=dsolve('Df=3*f+4*g,Dg=-4*f+3*g','f(0)=0,g(0)=1')
```

step 2 查看上面微分方程组的求解结果

```
>> disp('f=');disp(f)
f=
exp(3*t)*sin(4*t)
```

```
>> disp('g=');disp(g)
g=
[illegible]
```

[illegible]

5.11 利用 Maple 的资源

在 Maple 软件中，有 2000 多条符号计算命令。前面已经多次使用 Maple 的相关命令，但是这些都只涉及最常用的命令。在 MATLAB 中，[illegible] 利用 Maple 软件的符号计算[illegible]，[illegible] 多种 MATLAB 与 Maple 之间的接口，[illegible]。[illegible] 直接调用 Maple 的计算内核，在 Maple 中的某个命令，[illegible] 函数等。下面[illegible]。

5.11.1 调用 maple 的相关命令

在 MATLAB 中，[illegible] Maple 软件中[illegible]，[illegible]，使用应[illegible]，调用相应的相关命令。

例 5.52 在 MATLAB 中求解数值的阶乘。

step 1 在 MATLAB 中的命令窗口中输入下面的命令

```
>> kfrac=sym('k!');
>> syms k n
>> nfrac=subs(kfrac,k,n)
```

step 2 查看程序完成的结果

```
>> nfrac
 nfrac =
 n!
```

[illegible]，如果要在 MATLAB 中调用 Maple 的阶乘命令时，不能直接调用该命令，而必须调用 MATLAB 中相关的符号命令来要相引用该命令。

step 3 求出一个给定的计算数值，[illegible]

```
>> fivefrac=subs(kfrac,k,5)
   fivefrac =
   [illegible]
```

例 5.53 在 MATLAB 中调用 maple 中的命令，求解[illegible]

step 1 求出 maple 中的命令求解[illegible] $\begin{cases} y(n)*y(n-1)+y(n)-y(n-1)=0 \\ y(0)=a \end{cases}$ 的通解，在 MATLAB 的命令窗口中输入下面的内容

```
>> R=maple(' rsolve({y(n)*y(n-1) + y(n) - y(n-1) = 0, y(0)=a}, y)');
>> clR=class(R);
```

在上面的程序中，使用maple命令直接调用rsolve命令，求解上面的递推方程，然后使用class命令查看R变量的类别。

step 2 在命令窗口中输入下面的内容，查看上面命令得到的结果

```
>> R
R =
[illegible]
[illegible]
[illegible]
class =
char
```

从上面的结果不难看出，通过maple命令在MATLAB可以求解递推方程的结果，而且这结果的数据类型是"字符串"，而不是符号对象。

step 3 将上面的结果转换为符号对象，并修改其显示效果

```
>> M=sym(R);
>> pretty(M)
                                        a
                                    -------
                                    1 + 5 a
```

step 4 利用maple中的命令求解递推方程 $\begin{cases} F(n)=F(n-1)+F(n-2) \\ F(1)=1, F(2)=1 \end{cases}$ 的通解，在MATLAB的命令窗口中输入下面的内容：

```
>> S=maple('rsolve({F(n) = F(n-1) + F(n-2), F(1..2)=1},F)');
>> Sr=sym(S);
>> pretty(Sr)
                    1/2                  1/2 n          1/2                  1/2 n
        - 1/5 5     (1/2 - 1/2 5    )  + 1/5 5     (1/2 + 1/2 5    )
```

在上面的结果中，采用"F(1..2)=1"来表示初值条件F(1)=1，F(2)=1，这种表述方式是在使用maple函数时的用法。

step 5 查看该表达式的初值条件，在MATLAB的命令窗口中输入下面的内容

```
>> syms n
>> F1=subs(Sr,n,1);
>> F2=subs(Sr,n,2);
>> [F1;F2]
ans =
     1
     1
```

在上面的程序中，将符号表达式Sr中的n替换为1和2，就相当于求解表达式的初值条件F(1)和F(2)。得到的结果都是1。可见表达式符合初值条件。

查看 maple 的帮助

上面已经介绍如何在MATLAB中调用maple中的命令，限于篇幅，不能介绍maple中的所有命令。如果需要了解某个命令在maple的使用方法，则需要使用到maple的帮助。本小节中，将介绍如何在MATLAB中获得maple的帮助。

例 5.54 在 MATLAB 环境中查阅 maple 的各种帮助文件。

step 1 查看关于命令"mhelp"的帮助，输入"mhelp"，然后按"Enter"键，得到的结果如下：

```
>> mhelp
 MHELP  Maple help.
    MHELP topic prints Maple's help text for the topic.
    MHELP('topic') does the same thing.
    MHELP is not available with MATLAB Student Version.
    Example:
       mhelp gcd
    Reference page in Help browser
       doc mhelp
```

step 2 查阅maple在线帮助的索引条目，输入"mhelp index"命令，然后按"Enter"键，得到的结果如下：

```
>> mhelp index
 Index of help descriptions
Calling Sequence
     ?index[category]    or    help(index, category)
Description
- The following categories of topics are available in the help subsystem:
    expression  operators for forming expressions
    function    list of Maple functions
    misc        miscellaneous facilities
    module      topics related to modules
    packages    descriptions of library packages
    procedure   topics related to procedures and programming
    statement   list of Maple statements
  To access these help pages, you must prefix the category with index,
thus ?
  index[category] .
```

step 3 查阅maple中的具体分类目录，输入"mhelp index[function]"命令，然后按"Enter"键，得到的结果如下：

```
>> mhelp index[function]
 Index of descriptions for standard library functions
Description
- The following are the names of Maple's standard library functions.
For more
  information, see ?f where f is any of these functions.
    about              abs              add              addcoords
    additionally       addproperty      addressof        AFactor
    AFactors           AIrreduc         AiryAi           AiryAiZeros
    AiryBi             AiryBiZeros      algebraic        algsubs
    bernstein          BesselI          BesselJ          BesselJZeros
```

```
.....................................................................................
     Svd                  symmdiff          symmetric        syntax
     system               table             tan              tanh
    verify              WeberE           WeierstrassP    WeierstrassPPrime
     WeierstrassSigma     WeierstrassZeta   whattype         WhittakerM
     WhittakerW           with              worksheet        WRAPPER
     writebytes           writedata         writeline        writestat
     writeto              zero              Zeta             zip
     ztrans               _
See Also
libname, index[ package]
```

step 4 查看具体函数的帮助文件，在 MATLAB 中输入 "mhelp rsolve"，然后按 "Enter" 键，查看查询结果。

```
>> mhelp rsolve
 rsolve - recurrence equation solver
Calling Sequence
     rsolve(eqns, fcns)
     rsolve(eqns, fcns, 'genfunc'(z))
     rsolve(eqns, fcns, 'makeproc')
Parameters
     eqns - single equation or a set of equations
     fcns - function name or set of function names
     z    - name, the generating function variable
Description
- The function rsolve attempts to solve the recurrence relation(s)
specified in eqns for the functions in fcns, returning an expression
for the general term of the function.
.....................................................................................
Examples
> rsolve(f(n) = -3*f(n-1) - 2*f(n-2), f(k));
                                        k                          k
                  (2 f(0) + f(1)) (-1)   + (-f(0) - f(1)) (-2)
.....................................................................................
See Also
asympt, dsolve, genfunc, msolve, solve
```

从上面的查看结果中可以看出，关于一般函数的帮助信息包括函数的表达式、参数说明、描述和函数实例等，可以查阅其他函数的帮助信息。

可视化符号分析

在MATLAB中，数学工具箱为符号函数的可视化提供一组简易的命令，本节将着重介绍两个数学分析的可视化界面：单变量函数分析界面和泰勒级数逼近分析界面。其中，单变量函数分析界面由命令funtool引出，泰勒级数逼近分析界面由命令taylortool引出；引出界面之后，后续的所有操作都可以直接在界面上进行，下面分别详细介绍。

单变量函数分析界面

在MATLAB中，单变量函数分析界面主要用来分析单变量函数的关系，最多可以分析两个函数之

[illegible]关系，对于一些使用计算器或者只做一些简单的符号计算和图形处理的读者，该分析界面是一个很好的选择。该计算器功能简单，操作十分方便，一目了然。其调用格式为：funtool。当用户输入该命令后，MATLAB 会弹出如图 5.4 所示的界面。

图 5.4 单变量函数分析的默认界面

上面的界面是 MATLAB 调出单变量函数分析的默认界面。可以看出，在默认情况下，函数 f=x，g=1，[illegible]自变量 x 的取值范围是 [-2*pi, 2*pi]，常数 a=1/2。[illegible]可以看出，这个函数界面由两个子界面，即一个函数运算控制窗口和一个绘图窗口组成。在任何时候，两个图形窗口中有一个处于激活状态，由函数运算控制窗口上的操作只能对被激活的函数图形窗口起作用。

[illegible]使用该函数界面，在本小节中将修改 f(x) 的函数表达式，查看各种操作引起的图形的变化。修改后的界面如图 5.5 所示。

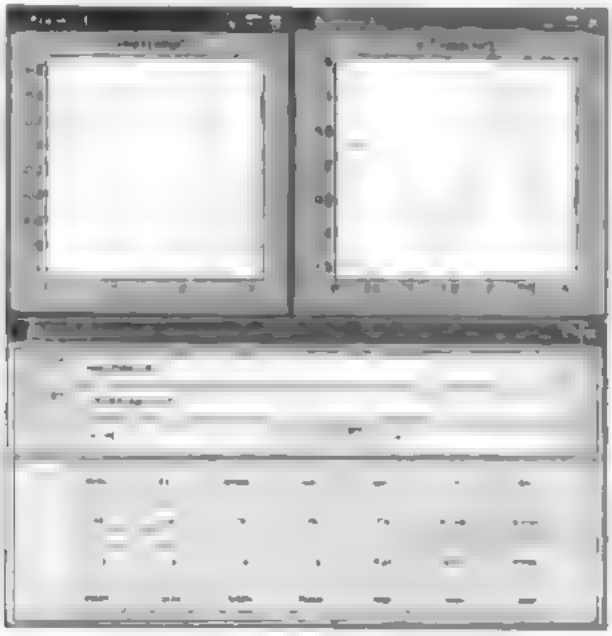


图 5.5 修改函数表达式和范围

在函数操作界面中，第一排的按钮都是针对函数f的，其具体含义分别是导入函数“diff f”按钮、积分“int f”按钮、简化“simple”按钮、提取系数“coeff”按钮、提取分数表达式的分子“numer”按钮、提取分数“denom”按钮、求反函数“inv”按钮等。

第二排的按钮主要用来处理函数f和系数a之间的运算，包括加、减、乘、除等，对应于按钮上的符号表示对应的操作。

第三排的按钮主要用来处理两个函数f和g之间的运算，其中按钮的功能可以直接从按钮的符号“猜到”，最后一个按钮“Swap”的功能是实现两个函数的转换。

最后一排的按钮主要用来进行一些辅助的操作，下面详细介绍各个按钮的具体功能。

- **Insert**：将当前编辑窗口中的函数插入函数列表fxlist中。
- **Cycle**：依次循环显示fxlist中的函数。
- **Delete**：从fxlist列表中删除当前编辑窗口中的函数。
- **Reset**：将计算器恢复到初始状况。
- **Help**：关于该界面的所有帮助内容。
- **Demo**：关于该界面的演示内容。
- **Close**：关闭该界面的所有窗口。

最后，该界面实际上是一个已经制作好的GUI界面，如果读者想查看其源程序，或者在其基础上修改或增强，可以首先查看其源程序。单击界面中的“Help”按钮，MATLAB会调用相关的命令，然后单击“View source for funtool”选项，打开该GUI界面的M文件，如图5.6所示。

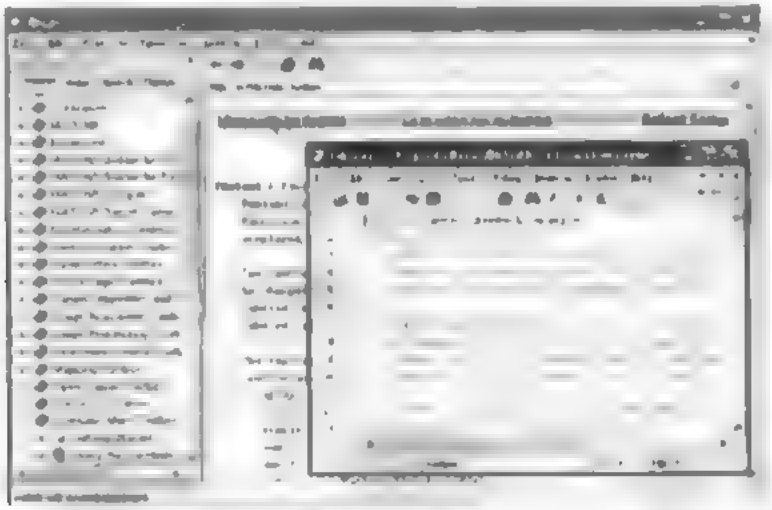


图5.6 查看GUI界面的M源程序

根据界面内容的不同，本界面的功能有限，由于本界面中的按钮功能针对函数f的较多，针对函数g的较少，但是可以使用“Swap”按钮来交换两个函数的位置。

5.12.2 泰勒级数逼近分析界面

在MATLAB的符号工具箱中，为用户提供“taylor”命令来求解符号表达式的泰勒级数展开式，使用该命令可以求解任意符号表达式在任意数据点的泰勒级数展开式。

泰勒级数逼近是一种十分重要的函数分析方法，在工程实践中要经常分析某个函数的函数形态，

因此，如要能够绘制出应呈现的函数图形，可以要求自动地计算函数的参数级数展开，应呈现函数的性态。

为了满足上面的条件，MATLAB提供参数级数展开分析界面，通过该界面的命令taylortool，用户在命令窗口中输入该命令后，MATLAB会自动调用界面，如图5.7所示。

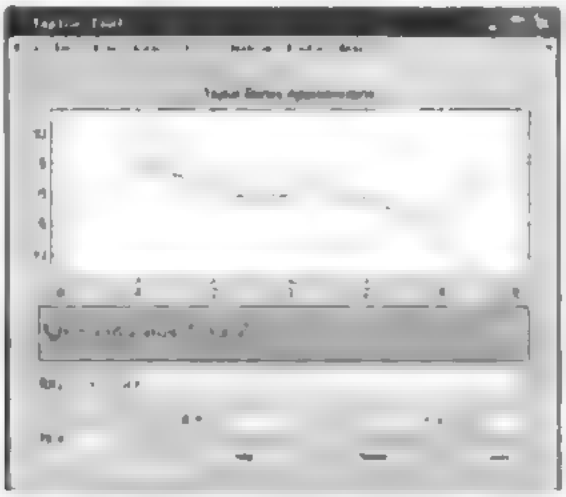


图5.7 参数级数分析的默认界面

在上面的界面中，a表示级数的展开点，N表示级数展开阶数，可以通过右侧的滚轮来改变数值，也可以在该栏中输入除整数值。函数级数展开的默认范围是(-2π,2π)，可以在相应的选项中修改范围的数值。

可以修改级数展开的所有参数，查看界面的相应变化，如图5.8所示。

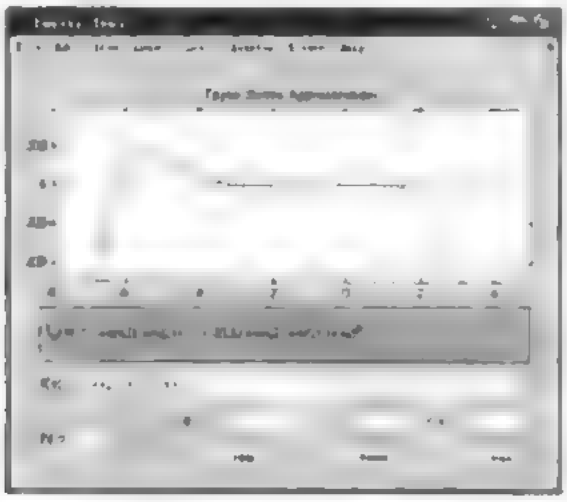


图5.8 对不同函数进行参数展开

5.13 小结

本章向读者介绍了关于符号计算的详细内容，包括符号对象、表达式、符号操作、符号代数方程、符号微分方程等。了解了这些内容后，基本上可以对MATLAB符号计算有一个大概的了解。在后面的章节中，将会继续向读者MATLAB中数据和函数的可视化。

第6章 数据和函数的可视化

本章包括

- 绘制二维曲线
- 绘制三维曲线图
- 绘制特殊图形
- 编辑三维图形的属性
- 图形的打印和输出
- 编辑二维曲线的属性
- 绘制三维曲面图
- 绘制四维图形
- 绘制复数图形

在前面的章节中，已经介绍和分析了MATLAB在数据处理、运算和分析中的各种运用。和其他科学计算工具类似，MATLAB也提供强大的图形编辑功能。通过图形，用户可以直观地观察数据间的内在关系，也可以十分方便地分析各种数据结果。从最初的版本开始，MATLAB就一直注重数据的图形表示，而且在更新版本的时候不断地使用新技术来改进和完善可视化的功能。

在本章中，将详细介绍MATLAB中的图形形成原理，曲线、曲面绘制的基本技巧，如何编辑图形参数，如何标记图形等。这些操作大部分只涉及MATLAB中的高层命令，这些命令格式简单，容易理解。对于底层的图形操作，将在后面章节中详细介绍。

最后，相对于MATLAB的6.x版本，MATLAB7提供功能十分强大、使用非常方便的图形编辑功能，使得原来十分复杂的图形编辑功能变得十分简单。这些内容将在本章的最后部分加以介绍。

图形的基础知识

在正式学习和熟悉MATLAB的图形编辑功能之前，有必要了解MATLAB中图形绘制的基本原理和基础步骤。本节大致将MATLAB图形的数据源分成离散和连续两个部分，介绍在MATLAB中如何绘制这两种图形。由于是介绍基础知识，所以选取的函数形式比较简单，主要是为了介绍绘制图形的原理和步骤。

离散数据（函数）的可视化

在MATLAB中，绘制离散数据的原理十分简单。对于离散数据数组$x=[x_1, x_2, x_3, ..., x_n]$，$y=[y_1, y_2, y_3, ..., y_n]$，MATLAB会将对应的向量组$(x_1,y_1)$，$(x_2,y_2)$，…，$(x_n,y_n)$用直角坐标中的点序列表示。一般情况下，MATLAB中的离散序列所反映的只是某确定区间的对应关系（也就是函数关系），不能反映无限区间上的对应关系。

例6.1 绘制离散函数$y=\frac{1}{(n-3)^2+1}+\frac{1}{(n-9)^2+4}$的图形，其中自变量$n$是取值范围是（0,16）的整数。

step 1 在MATLAB的命令窗口中输入下列内容：

```
>> n=0:16;
>> y = 1 ./ ((n-3).^2 + 1) + 1 ./ ((n-9).^2 + 4) + 5;
>> plot(n,y,'mh','markersize',15);
```

```
>> axis([0,17,5,6.2]);
>> grid on
```

step 2 按"Enter"键，MATLAB[illegible]

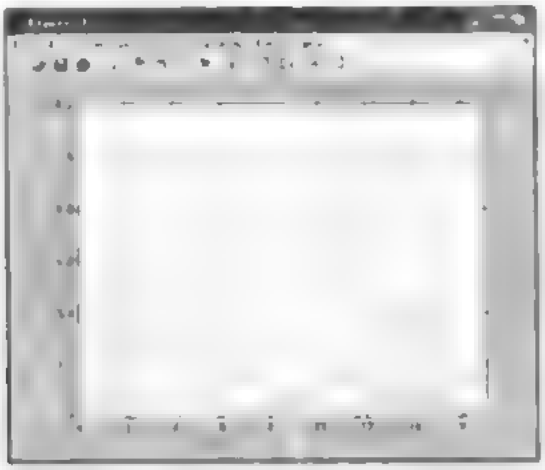


图 6.1 离散函数的可视化

[illegible]句的含义。

- [illegible]'markersize'[illegible]。其中，第一个参数'r'的[illegible]，m表示数字是"偶"，k表示数据点[illegible]。第[illegible]参数'markersize'中来设置标记的大小属性。15就是属性值。
- [illegible]。其中，横坐标轴（x轴）[illegible]纵坐标轴（y轴）的[illegible]，在MATLAB中对坐标轴的其他设置将在后面章节中详细介绍。
- [illegible]，MATLAB[illegible]轴和在x轴[illegible]。需要使用[illegible]，将在[illegible]的章节中介绍。

[illegible]，MATLAB[illegible]"[illegible]"的[illegible]，[illegible]MATLAB[illegible]。[illegible]特别要在命令窗口中显示，[illegible]

6.1.2 连续函数的可视化

在MATLAB中，[illegible]基本和离散函数相同，也就是在[illegible]，计算[illegible]函数值，然后将这[illegible]。[illegible]，没有太大的区别。在MATLAB中，将这些离散数据点转化为连续函数的方法是[illegible]。离散数据点的[illegible]，增加更多的数据点，计算各个数据点的函数数值，这个[illegible]。[illegible]直接将相邻的数据点用直线连接起来，用线性关系来替代其他的函数关系。

在MATLAB中，使用上面两个函数可以绘制连续函数的图形。如果函数的自变量采样点数不够多，则绘制的连续曲线就会产生失真现象。

例6.2 绘制函数 $y=\frac{1}{(x-3)^2+1}+\frac{1}{(x-9)^2+4}+5$ 的图形，其中自变量的取值范围是 $[0,16]$ 的所有实数。

step 1 在MATLAB的命令窗口中输入下列内容

```
>> x1=0:16;
>>y1 = 1 ./ ((x1-3).^2 + 1) + 1 ./ ((x1-9).^2 + 4) + 5;
>> x2=0:0.02:16;
>>y2 = 1 ./ ((x2-3).^2 + 1) + 1 ./ ((x2-9).^2 + 4) + 5;
>> subplot(2,2,1),plot(x1,y1,'rp'),axis([0,16,5,6.[illegible]),title('Picture 1'),
grid on
>> subplot(2,2,2),plot(x2,y2,'rp'),axis([0,16,5,6.[illegible]),title('Picture 2'),
grid on
>> subplot(2,2,3),plot(x1,y1,x1,y1,'rp'),axis([0,16,5,6.[illegible]),title
('Picture 3'),grid on
>> subplot(2,2,4),plot(x2,y2,'LineWidth',2),axis([0,16,5,6.[illegible]),title
('Picture 4'),grid on
```

step 2 按"Enter"键，就可以得到相应的结果，如图6.2所示。

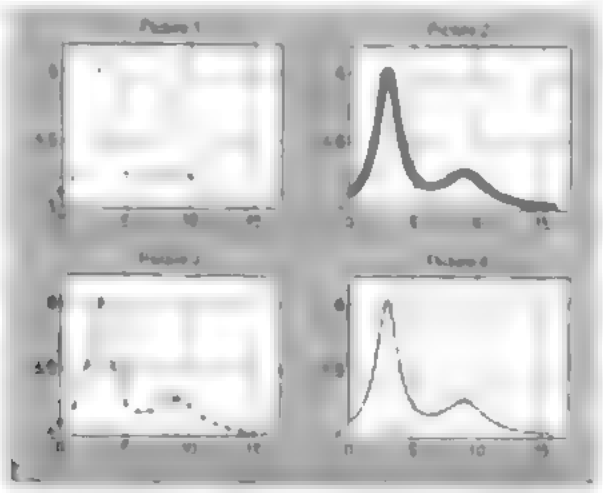


图6.2 连续函数的表示方法

step 3 分析图形结果。

上面的图表命令中，Picture1和Picture3使用的是数组x1和y1，Picture2和Picture4使用的是数组x2和y2。从上面的对比[illegible]保持宜的看出，当在多维使用的数组非常密集的情况下，图表可以更明显地表示出函数的形态。

Picture 1和Picture 3虽然也表现出了大概的形态，但是在图表的拐角处明显的失真，不能很好表示函数性态。

从上面的程序中可以看出，对于离散数据，MATLAB的plot命令在默认情况下会自动将这些离散数据用直线连接起来，使之成为连续曲线。

[illegible]

6.1.3 绘制图表的基础步骤

[illegible] MATLAB 中绘制图表的基本步骤如下。

step 1 准备要绘制的数据。这是基础的步骤，[illegible]

step 2 设置[illegible]

step 3 绘制，并设置相应的参数。这是在 MATLAB 中绘制图表的主要步骤，绘图命令数量[illegible]

step 4 设置坐标轴属性。从这个步骤开始，开始对图表的编辑[illegible]。用户可以设置坐标轴的多[illegible]

step 5 [illegible]

[illegible]

6.2 绘制二维曲线

[illegible]

6.2.1 plot 命令

[illegible]

在 MATLAB 中，调用 plot 的方法有下面 3 种。

- plot(X,'PropertyName',PropertyValue,...)

  [illegible]

型、颜色和数据点形等；PropertyValue表示对应属性的选值。对于参数X的不同类型，MATLAB会作不同的处理，详细内容如下：

- 当X是实数向量时，MATLAB会以X向量元素的下标为横坐标，元素数值为纵坐标绘制连续曲线；
- 当X是实数矩阵时，MATLAB会绘制矩阵中每列数值元素相对于其下标的连续曲线，因此绘制的图表结果中包含多个图表，个数等于X矩阵的列数；
- 当X是复数矩阵的时候，以列为单位，分别以矩阵元素的实部为横坐标，以元素的虚部为纵坐标绘制连续曲线。

◆ plot(X,Y,'PropertyName',PropertyValue,...)

和上面的命令相比，该命令多了一个参数Y，其中X，Y都是图表的数据数组。其他参数的含义和上面的命令相同。同样，对于参数X、Y的不同数据类型，MATLAB也会作不同的处理，详细内容如下：

- 当X、Y是同维向量时，MATLAB会以X和Y元素为横、纵坐标绘制曲线；
- 当X是向量，Y是某维数值与X向量相同的矩阵时，MATLAB会绘制多个连续曲线，默认情况下这些曲线的颜色都会不同；曲线的个数等于Y矩阵的另外一个维数，X向量是曲线的横坐标。
- 当X是矩阵，Y是向量时，情况和上面部分相同，Y向量是这些曲线的纵坐标。
- 当X、Y是同维矩阵时，MATLAB会以X、Y对应元素为横、纵坐标绘制曲线，因此，曲线的个数就是矩阵的列数。

◆ plot(X1,Y1, X2,Y2, X3,Y3,...,'PropertyName',PropertyValue,...)

该命令和前面的命令plot(X,Y,'PropertyName',PropertyValue,...)相似，只是同时在图形窗口中绘制多条曲线，这些曲线之间没有相互的约束。

plot命令的实例

例6.3 演示下面的案例，来了解上面plot的二个调用形式。

step 1 在MATLAB的命令窗口中输入下列内容：

```
>> k=0.2:0.2:1;
>>t=0:0.01:1;
>>y=exp(t')*k;
>> subplot(2,2,1),plot(y),axis([ 0,100,0,3] ),title('Picture 1'),xlabel
('下标'),ylabel('y')
>> subplot(2,2,2),plot(t,y),axis([ 0,1,0,3] ),title('Picture 2'),xlabel
('t'),ylabel('y')
>> subplot(2,2,3),plot(y,t),title('Picture 3'),xlabel('y'),ylabel('t')
>> subplot(2,2,4),plot(t),title('Picture 4'),xlabel('下标'),ylabel('t'),
axis([ 0,100,0,1] )
```

step 2 按"Enter"键，就可以得到相应的结果，如图6.3所示。

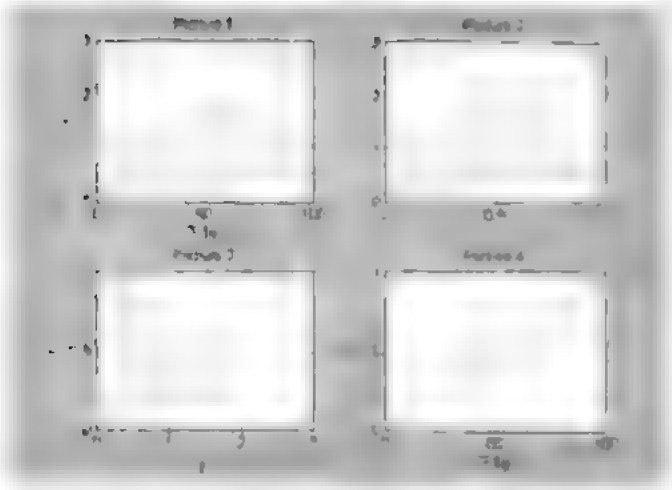


图6.3 演示plot基础命令

step 3 分析图形结果。

在[illegible]的子图"[illegible]"中，使用的命令是"plot(y)"。其中参量y是一个[illegible]的矩阵，数值就是函数y=e^x×x的计算结果。根据前面的结果，MATLAB会绘制5组"[illegible]"的函数曲线，而曲线以矩阵的元素下标为单位，因此，最大横坐标为100。从结果图可以看出，这5条曲线的颜色是各异的。

在子图"[illegible]"中，绘图命令是"plot(t,y)"。由于t是一个100维的行向量，y是一个100×5的矩阵，因此，MATLAB会以t横坐标和纵坐标绘制5条颜色各异的曲线。

在子图"[illegible]"中，使用的命令是"plot(y,t)"。其情况和上面的子图类似，只是这5条曲线都会以t为共同的纵坐标。

在子图"[illegible]"中，使用的命令是"plot(t)"。由于t是一个100维的行向量，因此，MATLAB会绘制以下标为横坐标的元素曲线。

[illegible]

例6.4 在MATLAB中绘制复数函数$f(t)=e^{-t}(t+\sin(2t)+3i\cos t+\log(1+t))$的曲线，并在曲线上显示对应的坐标。

step 1 在MATLAB的命令窗口中输入下述语句：

```
>> t=[0:0.1:25]';
>>my_medium=[t,[illegible]];
>> plot(my_medium,'Linewidth',2)
>>[illegible]
>> xlabel('t'); ylabel('Y-axis')
```

step 2 按"Enter"键，显示如图6.4所示的结果，[illegible]。

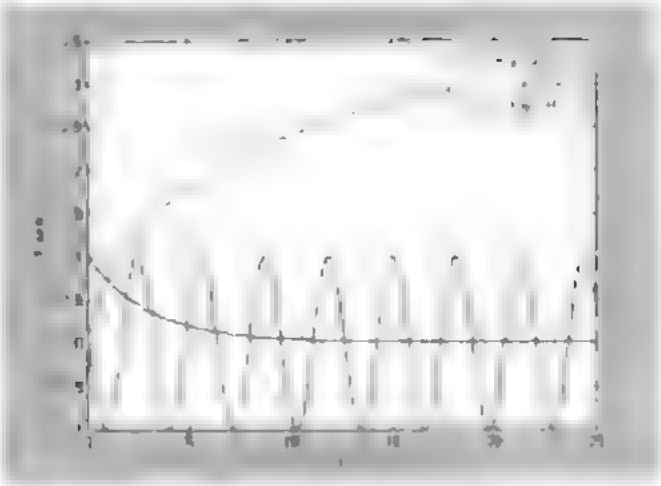


图6.4 绘制复数矩阵

在上面的实例中，复数矩阵和一个实数矩阵相加，以产生新的矩阵，将一组的曲线绘制在同一个坐标系中，并使用不同的颜色来表示，使用命令 legend 添加了图表的图例。

提示

从上面的实例中可以看出，在 MATLAB 环境下，不同的曲线命令十分简单，只需要将函数表达式写成矩阵形式，直接使用 plot 命令就可以实现。

6.2.3 曲线的颜色、线型和数据点形

为了能使曲线更直观，在绘制多条曲线时，为了将各个数据系列，在 MATLAB 中，用户可以为曲线设置不同的颜色、线型和数据点形属性。

在 MATLAB 中，关于曲线线型和颜色参数的设置如表 6.1 所示。

表6.1 MATLAB 中线型和颜色的设置

项目	符号	含义
线型	-	实线
	:	虚线
	-.	点划线
	--	长划线
颜色	b	蓝
	g	绿
	r	红
	c	青
	m	品红
	y	黄
	k	黑
	w	白

这样，将曲线线型符号和颜色一起写入，导致在使用 plot 命令时出现歧义，参数冲突。为使用，例如，'r-' 表示曲线是红色的虚线。在命令 plot(X,'Property1','Property1Value',...) 中，如果没有输入参数 'Property1Name' 和 'Property1Value' 的数值，MATLAB 会将线型和颜色的属性设置为默认值。线型的默认值是 '实线'，颜色的默认值则按次序是蓝、绿、红、青等，如果只有一条曲线，则线的颜色为蓝色。

色的。

[illegible]

在MATLAB中，用户能[illegible]曲线属性[illegible]，除此之外，[illegible]曲线中的数据点设置属性，这样，用户[illegible]数据点形，[illegible]曲线[illegible]。MATLAB中的数据点形的属性如表6.2所示。

表6.2 MATLAB中的数据点形

符号	含义	符号	含义
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]

例6.5 [illegible]

step 1 在MATLAB命令窗口中输入下列语句

```
>> x = 0:pi/8:4*pi;
>>y = exp(-x/3).*sin(3*x);
>>yb=exp(-x/3);
>>plot(x,yb,'-k',x,-yb,'-k',x,y,'-.ro','LineWidth',2,...
                    'Ma[illegible]','[illegible]',...
                    'Ma[illegible]','y',[illegible]
                    'Ma[illegible]',6)
>>grid on
```

step 2 [illegible]，如图6.5所示。

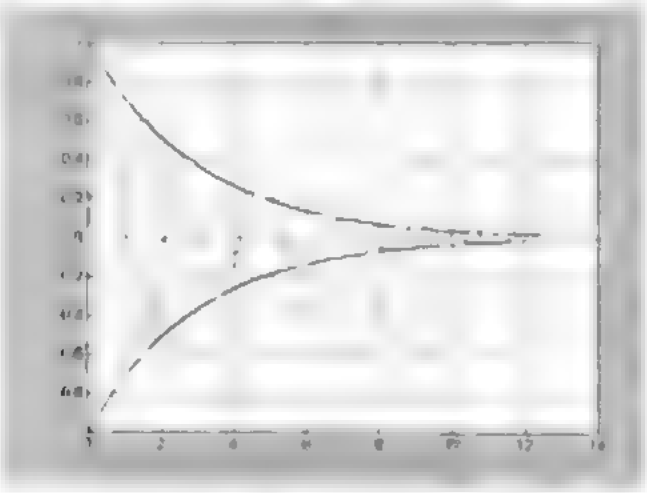


图6.5 设置曲线属性

step 3 分析上面的图形结果。

在上面的图表绘制过程中，只使用了一个命令"plot(x,yb,'-k',x,yc,'-k',x,y,'-ro','LineWidth',2,'MarkerEdgeColor','g','MarkerFaceColor','y','MarkerSize',6)"就将上面的图表绘制完成。

其中，yb和yc数据系列是曲线的对称线，其曲线的属性是'-k'，也就是黑色的实线；数据y就是函数的曲线，其曲线的属性是'-ro'，即该曲线是红色带点实线，其数据点标记为空心圆圈。参数'MarkerEdgeColor'将空心圆圈的边框颜色设置为绿色(g)，参数'MarkerFaceColor'将空心圆圈的填充颜色设置为黄色(y)，参数'MarkerSize'将空心圆圈的大小设置为6。

> **说明** 从上面的例子可以看出，在MATLAB中只要通过简单的指令就可以设置图形的各种属性，包括线型、颜色和数据标记等参数，可以绘制出丰富的图形。

6.2.4 设置坐标轴范围

在绘制坐标轴时要根据图表的需要，选择不同的坐标轴范围。一般MATLAB会根据数据的范围给出适当的坐标轴默认设置。但是并不是所有情况下的默认设置都是最好的，用户可以根据需要手动设置坐标轴的属性。MATLAB提供一系列关于坐标轴的命令，用户可以根据自己的需要，调整坐标轴的标度、刻度、外观和纵横比等。表6.3列出了MATLAB中关于坐标轴的常用命令。

表6.3 MATLAB中的数据点形

命令	含义	命令	含义
axis auto	使用默认设置	axis manual	保持当前刻度范围
axis off	取消坐标轴背景	axis on	使用坐标轴背景
axis ij	原点在左上方	axis xy	原点在左下方
axis equal	横、纵坐标使用相同刻度	axis image	等长刻度，坐标框紧贴数据范围
axis normal	默认的矩形坐标系	axis square	正方形坐标系
axis tight	将数据范围设置为刻度	axis fill	使坐标充满整个绘图区

上表所示的命令中，主要是针对图表坐标轴的外观。本节着重介绍如何直接设置坐标轴范围的控制命令axis([xmin xmax ymin ymax])，来设置图表在坐标轴上的显示范围。在本小节中，将通过一个例子来说明如何在MATLAB中对图形的坐标轴进行控制。

例6.6 在MATLAB中绘制函数$y = \tan x$在$[0, \frac{\pi}{2}]$中的图表。

step 1 首先使用MATLAB默认的坐标轴范围绘制该图表，在命令窗口输入以下代码：

```
>> x = 0:.01:pi/2;
>>plot(x,tan(x),'-ro')
```

step 2 查看图形结果。按"Enter"键，就可以得到相应的图形，如图6.31所示。

step 3 修改坐标轴的范围。在MATLAB的命令窗口，输入下列代码：

```
>> axis([0  pi/2  0  50])
>> grid on
```

step 4 查看图形结果。按"Enter"键，就可以得到相应的图形，如图6.32所示。

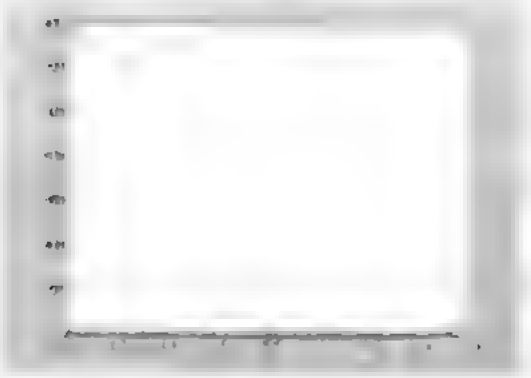


图6.6 绘制默认图形

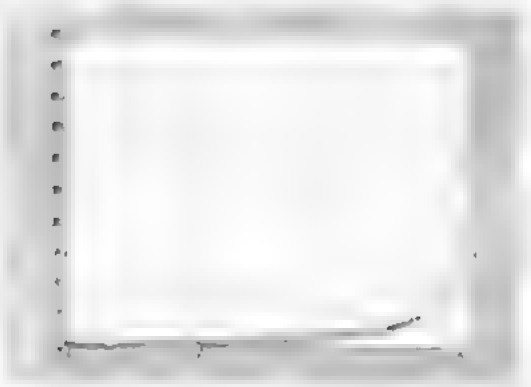


图6.7 修改后的图形

[illegible]

说明

在上述的中[illegible]

6.2.5 设置坐标轴显示方式

例6.7 [illegible]

step 1 在命令窗口输入以下内容

```
>> t=[0:pi/50:2*pi]';
>> x=2*[illegible]
>> y=4*sin(t);
>> [illegible]
>> [illegible]
>> [illegible]
>> [illegible]
>> [illegible]
```

```
Fill');
>> [illegible]
```

step 2 按 'Enter' 键，[illegible]

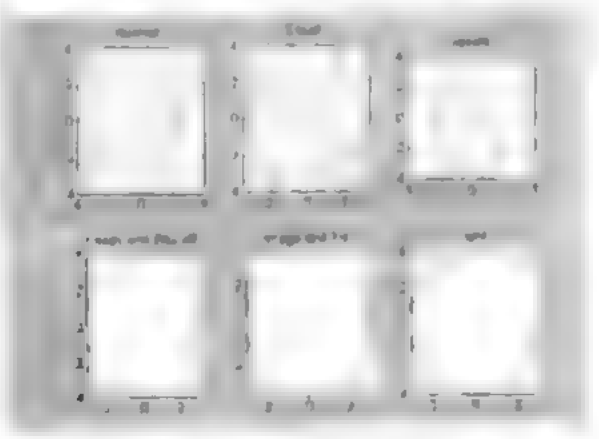


图6.8 不同的坐标轴显示方式

6.2.6 设置坐标轴系统

例6.8 [illegible]

step 1 在命令窗口输入以下内容

```
>> x=logspace(-2,0,500);
>>y=[illegible]
>>subplot(2,3,1), plot(x,y); Grid on;title('Normal');
>>subplot(2,3,2), plot(x,y);set(gca,'Xscale','log','Xdir','reverse');
Grid on;Title('Y-Logx');
>>subplot(2,3,4), plot(x,y);set(gca,'Xdir','reverse','Ydir','reverse');
Grid on;Title('LogY-X');
>>subplot(2,3,4), plot(x,y);set(gca,'Xdir','reverse','Ydir','normal');
Grid on;Title('Y-revX');
>>subplot(2,3,5), plot(x,y);set(gca,'XDir','normal','Ydir','reverse');
Grid on;Title('revY-X');
>>subplot(2,3,6), plot(x,y);set(gca,'Xscale','log','Yscale','normal');
Title('Y-LogX');set(gca,'Xdir','rev','Ydir','rev');
```

step 2 按 'Enter' 键，[illegible]

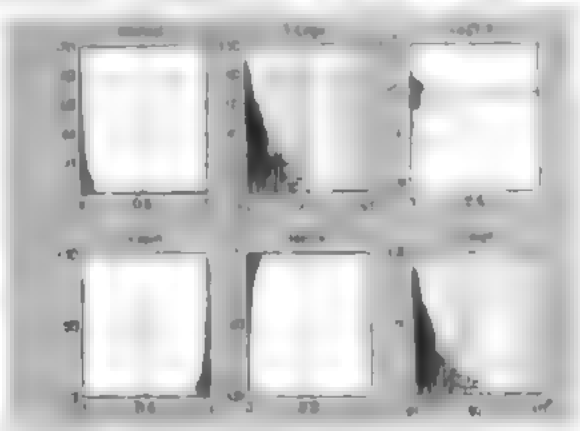


图6.9 不同的坐标轴系统

step 3 分析图形结果。

上面的语句中[illegible]，[illegible]

在"[illegible]"[illegible]，使用默认坐标轴系统，[illegible]，都在直角坐标系中，[illegible]管理采用MATLAB的默认设置。

在"[illegible]"[illegible]，将X坐标轴转化为log的对数坐标系，而Y坐标轴保持直角坐标系，而[illegible]这个操作涉及坐标轴的转换，[illegible]。在上面的程序中，使用时间参数语句[illegible]，将X坐标轴转换为log，而Y坐标轴保持默认的linear数值。在"[illegible]"[illegible]，将Y坐标轴转换为log)，而X坐标轴保持直角坐标系。

在"[illegible]"[illegible]，将X坐标轴倒置，Y坐标轴保持不变。[illegible]，这个操作涉及的[illegible]表坐标轴的转换，[illegible]。[illegible]，在"[illegible]"子图形中，将Y坐标轴倒置，X坐标轴保持不变。

在最后一个表示中，使用[illegible]属性，设置X坐标轴的分隔线，而将Y坐标轴的分隔线清除。

在上面的一系列操作中，[illegible]，[illegible]，[illegible]MATLAB的[illegible]，[illegible]，[illegible]，[illegible]。[illegible]

627 图形标识

在MATLAB中提供了[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]、坐标轴名称、图形[illegible]等。[illegible]，MATLAB提供了[illegible]。在本小节中，将使用一个简单的函数来说明[illegible]。

例69 在MATLAB中绘制函数$f_1(x)=-x^2+4x+450$和$f_2(x)=2^{-x}+5x$，[illegible]两个曲线的交点标曲线的属性。

step 1 打开M文件编辑器，在其中定义myfun函数，在编辑器中输入以下语句：

```
function f=myfun(x)
y1=-x.^2+4*x+450;
y2=2.^(-x)+5*x;
f=y1-y2;
```

上面的语句要在M文件编辑器中定义函数，[illegible]，将上面的函数保存为myfun.m文件，然后关闭文件编辑器。

[illegible]

step 2 在MATLAB的命令窗口中输入以下语句：

```
>>x=[-10:0.1:10];
>> y1=-x.^2+4*x+450;
>> y2=2.^(-x)+5*x;
```

```
>> plot(x,y1,x,y2,'Linewidth',2);hold on
>>xrt=fzero(@myfun,18);
>>yrt=-xrt^2+4*xrt+4;
>> plot(xrt,yrt,'r.','MarkerSize',20);hold off
>> text(xrt,yrt,' fontsize(16) leftarrow fontname(楷书) 交点')
>>annotation1 = annotation(...
 'textarrow',...
  [ 0.6625 0.5036],[ 0.2262 0.372],...
  'LineWidth',1,...
  'HeadStyle','plain',...
  'String',{'这是一个','递增函数'},...
  'FontName','Times New Roman',...
  'FontSize',9,...
  'HorizontalAlignment','left',...
  'TextBackgroundColor',[ 1 1 1],...
  'TextLineWidth',1,...
  'TextEdgeColor',[ 0 0 0]);
>>annotation2 = annotation(...
  'textarrow',...
  [ 0.2732 0.375],[ 0.5714 0.7143],...
  'String',{'这是一个','递减函数'},...
  'FontSize',9,...
  'TextEdgeColor',[ 0 0 0]);
>> xlabel('x')
>> ylabel('y')
>> title('两个曲线的交点')
```

Step 3 按"Enter"键，就可以得到如图6.10所示。

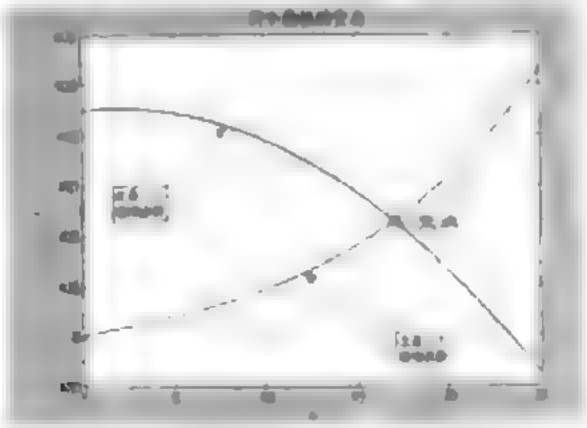


图6.10 完成的图形标识

Step 4 分析上面的图形结果。

在上面的图形中，为图形添加了多个图形标识：坐标轴、标题、文本注释、箭头注释等。其中，坐标轴和标题的设置方法十分简单，直接可由简单命令实现，这里就不介绍了，而文本注释和箭头注释稍微比较复杂，下面详细分析上面两种注释方式。

命令：'text(xrt,yrt,'fontsize(16) leftarrow fontname(楷书) 交点')'为两个曲线交点处添加文本注释"交点"，其中参数xrt和yrt表示插入文本注释的位置，由前面求解'fzero(xrt,yrt)'产生参数'fontsize(16) leftarrow fontname(楷书) 交点'定义了注释文本字体（楷书）和大小（16），以及文本内容（交点），其中参数leftarrow表示为文本添加左向箭头。

命令"annotation('textarrow',[0.6625 0.5036],[0.2262 0.372],'LineWidth',1,'HeadStyle','ellipse','Text','strong','这是一个"递增函数"','FontName','Fontti toc','FontSize',3,'VerticalAlignment','left','TextBackgroundColor',[1 1 1],'TextLineWidth',1,'TextEdgeColor',[0 0 0])"一个是一个以箭右箭头的图形注释。其中"[0.6625 0.5036],[0.2262 0.372]"表示添加箭头的两个坐标数值，其余的参数则是设置箭头注释文本的属性。

在上面的绘图命令中的"text"函数，可以使用MATLAB的图形对象属性来设置文本的属性，使图形的动态效果更加丰富多彩。

6.2.8 叠绘

在MATLAB中，用户可以使用多种命令在同一个图形窗口绘制不同图形，但是在某些应用中，用户会遇到在已经绘制了图形的基础上再绘制曲线的情况，为了达到这种效果，MATLAB系统向用户提供了hold命令。

在MATLAB中，hold命令的常用调用格式为

- hold on 启动图形保持状态，当前坐标轴和图形都将保持，此后绘制的图形都将添加在这个图形基础上，并且自动调整坐标轴的范围。
- hold off 关闭图形保持功能。
- hold 在hold on和hold off命令之间进行切换。

例6.10 在MATLAB中绘制函数 $f_1(x)=-x^2+4x+450$ 和 $f_2(x)=2^{-x}+5x$ 的曲线，然后将 $f_2(x)$ 向下平移50个单位。

step 1 在MATLAB的命令窗口中输入一段程序

```
>> x=[0:0.01:25];
>> y=-x.^2+4*x+450;
>> plot(x,y,'LineWidth',2);hold on
```

step 2 按"Enter"键，绘制第一个函数的曲线，如图6.11所示。

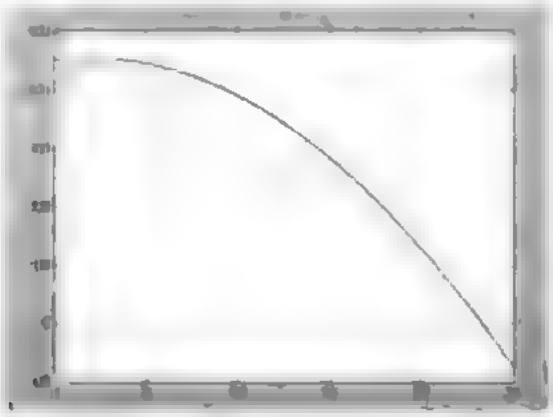


图6.11 第一个函数的曲线

step 3 在MATLAB的命令窗口中输入下面的内容

```
>> f=2.^(sqrt(3*x))+5*x;
>> plot(x,f,'LineWidth',2);
```

step 4 查看绘图结果。输入代码后，按"Enter"键，绘制的曲线如图6.12所示。

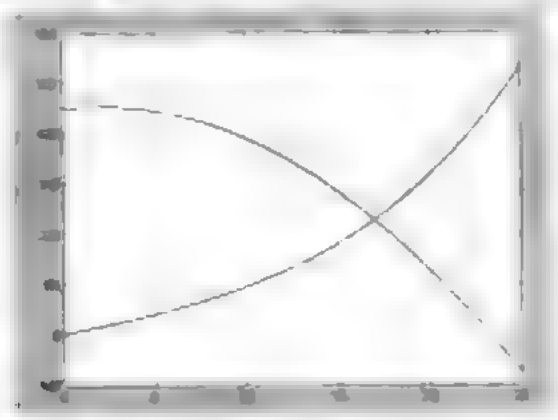


图6.12 添加第二个函数的曲线

step 5 在MATLAB的命令窗口中输入下面的内容

```
>> plot(x,f-50,'LineWidth',2);
>> hold off;
```

step 6 查看绘图结果。输入代码后，按"Enter"键，绘制的曲线如图6.13所示。

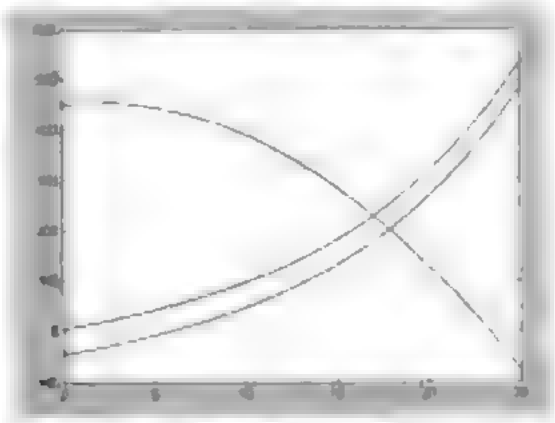


图6.13 添加第三个函数的曲线

step 7 最后，在MATLAB的命令窗口中输入下面的内容

```
>> plot(x,f-50,'LineWidth',2);
```

step 8 查看绘图结果。输入代码后，按"Enter"键，绘制的曲线如图6.14所示。

[illegible]

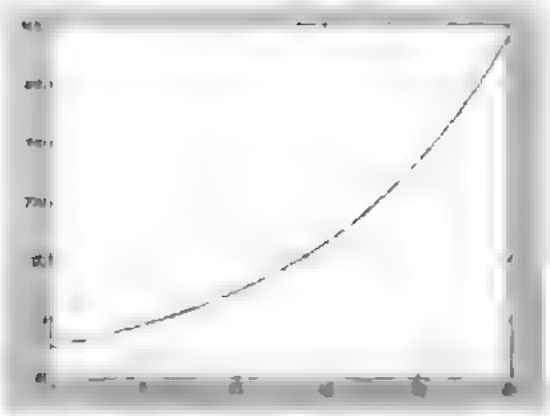


图6.14 单独绘制最后一个函数的曲线

6.2.9 双坐标轴

[illegible]，MATLAB提供plotyy命令。

plotyy命令的常用调用格式如下

- [illegible]
- [illegible]指定
- [illegible]曲线类型由FUN1指定，x2-y2曲线类型由FUN2指定。

[illegible]

例6.11 [illegible]Profit_Margin的数据[illegible]要在MATLAB中实现上面的要求。

step 1 在MATLAB命令窗口中输入下面的内容

```
>> Income=[2456,2032,1900,2450,2890,2280];
>> Profit_Margin=[12.5,11.3,10.2,14.5,14.3,15.1]/100;
>> t=1:6;
>> [AX,H1,H2] = plotyy(t,Income,t,Profit_Margin,'bar','plot');
```

step 2 查看[illegible]

step 3 然后在MATLAB命令窗口中输入下面的内容

```
>>set(get(AX(1),'Ylabel'),'String','Left Y-axis');
>>set(get(AX(2),'Ylabel'),'String','Right Y-axis');
[illegible]
```

```
'Marker','o',...
'MarkerEdgeColor',[1 0 0],...
'MarkerFaceColor',[1 0 0]);
[illegible]
[illegible]
'FaceColor',[1 1 0.4745]);
```

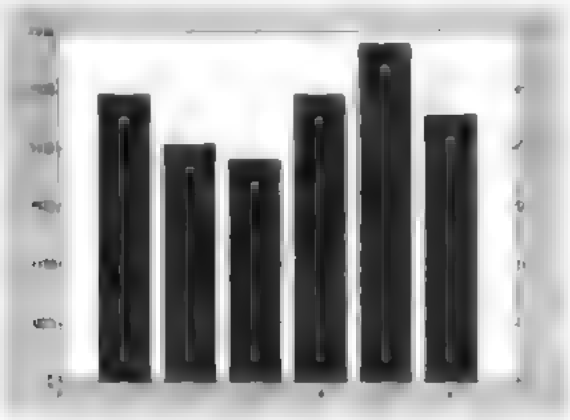


图 6.15　绘制的基础双轴图形

Step 4 高亮[illegible]，输入[illegible]，按"Enter"键，[illegible]。

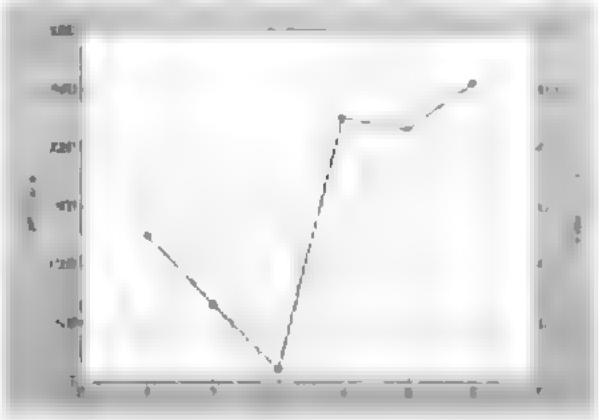


图 6.16　经过编辑后的曲线

Step 5 分析上面的图形结果。

在上述的步骤[illegible]中，使用plot()函数绘制了基础的点线曲线，对于第一个数据系列，Excel使用的是表示为虚线的blue，而第二个数据系列Period，Marker使用了表示为实线的plot。在[illegible]，MATLAB的绘图函数[illegible]的属性都设置了默认值，你可以[illegible]。

在步骤[illegible]中，通过使用图形句柄[illegible]，你可以修改、填充新的数据时，[illegible]句柄的内容，这些内容将在后面的章节中加以详细介绍。

[illegible]legend()[illegible]

6.2.10 多子图

在前面一节中，一般多次调用在一个图形窗口中在重复个独立的子窗口的情况。在[illegible]表示窗口的[illegible]个数，[illegible]使用[illegible]。在 MATLAB 中，提供 subplot 命令来绘制子图。

subplot 命令的常见调用格式如下

- subplot(m,n,p)　该命令的功能是将绘图窗口分成 m × n 个窗口，然后在第 p 个窗口中创建坐标轴，将其设置为当前窗口。
- subplot(m,n,p,'replace')　如果在绘制图形之前已经存在子坐标轴，该命令将删除重叠的坐标轴，创建一个新的坐标轴系统。
- subplot(m,n,p,'align')　该命令的功能是对齐坐标轴。
- subplot(h)　使句柄 h 对应的坐标轴成为当前坐标轴
- subplot('position',[left bottom width height])　在给定位置上创建子图，并将其设置成当前坐标轴。

在命令 subplot(m,n,p)中，[illegible]的子图的编号[illegible]是，[illegible]方是第[illegible]，[illegible]。[illegible]在命令 subplot('position',[left bottom width height])中，[illegible]，[illegible]。[illegible]subplot 命令生成的子图都是矩形区域的。

例 6.12　在 MATLAB 中，用子图绘制函数 $y_1=e^{-t/3}$，$y_2=\log(1+t)$，$y_3=\sin(2t+3)$ 的图形，分别使用不同的子图绘制命令。

step 1　在 MATLAB 的命令窗口中输入以下语句

```
[illegible];
>> y1=exp(-t/3);y2=log(1+t);y3=sin(2*t+3);
>> subplot(2,2,1),plot(t,y1,'Linewidth',2)
>> subplot(2,2,2),plot(t,y2,'Linewidth',2)
>> subplot('position',[0.2,0.05,0.6,0.45])
>>  plot(t,y3,'r','Linewidth',2)
```

step 2　按"Enter"键，[illegible]命令，[illegible]，如下图。

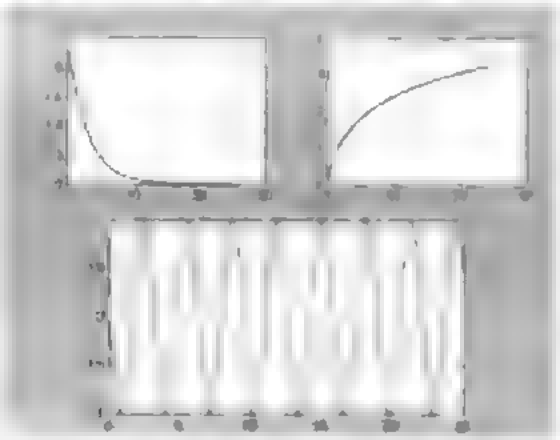


图 6.17　绘制多子图图形

说明 在上面的命令中，[illegible]

6.2.11 交互式图形命令

在 MATLAB 中，除了[illegible]之外，也提供[illegible]交互式图形命令，例如，[illegible]有数值，这个时候使用 ginput 命令[illegible]标值。

除了 ginput 命令之外，MATLAB [illegible] zoom 等命令，[illegible]命令中，除了 ginput 命令[illegible]其中 ginput 命令和 zoom 命令[illegible]使用，[illegible]数据。

在 MATLAB 中，ginput 命令的常用调用格式如下

- ◆ [x,y] = ginput(n)，该命令[illegible]，用 Enter 键来结束取点。
- ◆ [x,y] = ginput，取点的数目不受限制，这些数据[illegible]坐标数值保存在[x,y]中，通过 Enter 键来结束取点。
- ◆ [x,y,button] = ginput(...)，返回值 button [illegible]数据[illegible]

说明 [illegible]

在 MATLAB 中，ginput 命令[illegible]和 zoom 命令[illegible]，zoom [illegible]，[illegible]形的缩放，其常见的调用格式如表 6.3 所示。

表 6.3 MATLAB 中的 zoom 命令

命令	含义	命令	含义
zoom	[illegible]	zoom [illegible]	[illegible]轴的缩放
zoom on	允许对坐标轴进行缩放	zoom [illegible]	[illegible]坐标轴进行缩放
zoom out	恢复到坐标轴的设置	zoom reset	[illegible]
zoom [illegible]	[illegible]轴进行缩放	zoom [illegible]	[illegible]

说明 在 MATLAB 中，使用 zoom 命令[illegible] Windows [illegible]，[illegible]来缩放了。

6.2.12 交互式图形实例

例 6.13 在 MATLAB 图形窗口中[illegible]，[illegible]键

[illegible]，然后使用样条插值法在所有数据点连接起来，绘制出[illegible]

step 1 在MATLAB命令窗口中输入下列语句

```
>> axis([0 10 0 10])
>> hold on
>> xy = [];
>> n = 0;
>> disp('Left mouse button picks points.')
>> disp('Right mouse button picks last point.')
>> but = 1;
>> while but == 1
    [illegible]
    plot(x1,y1,'ro')
    [illegible]
    [illegible]
end
>> t = 1:n;
>> ts = 1: 0.1: n;
>> xys = spline(t,xy,ts);
[illegible]
>> hold off
```

step 2 查看[illegible]，输入[illegible]，按"Enter"键，MATLAB会[illegible]，使用鼠标左键选择数据点，然后点击鼠标右键，结束数据点选取，得到[illegible]形如图6.18所示。

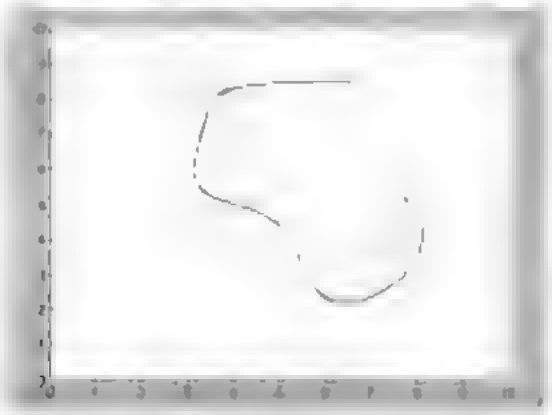


图6.18 绘制动态图形

step 3 重新选择数据，绘制新图形。[illegible]数据点[illegible]，数据有任何限制，[illegible]，现在需要随意选取一些点，而[illegible]重新选择数据，然后使用[illegible]，如图6.19所示。

[illegible]

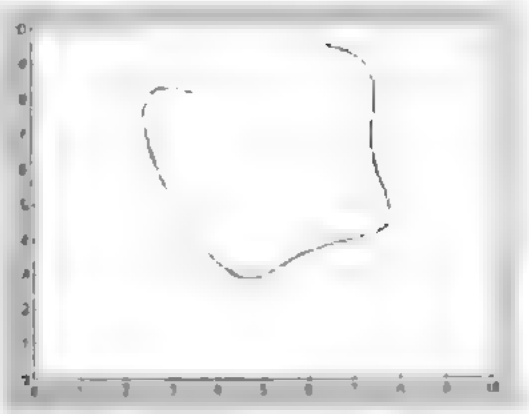


图6.19 修改动态图形

6.2.13 fplot 命令

在前面的小节中，读者已经对plot命令有了详细的了解。对于MATLAB对一维函数和二维函数提供了不同的绘图命令。其中比较常见的命令是fplot和ezplot命令，两种命令的适用范围不相同。

简单来说，前面介绍的plot命令是将函数值或者函数值的数值矩阵转化为连续图形。在实际应用中，如果人为了解算法函数随自变量变化的趋势，也使用plot命令对函数绘图，则有可能因为自变量的范围选取不当使函数绘图失真，可以根据需分析出来，将自变量的取值调整得小或大，但这种方法会增加MATLAB处理数据的负担，降低效率。

对于上面的问题，MATLAB提供了fplot函数来解决。该命令主要将给定的函数值的数值数据矩阵，全交由内部的自适应算法来决定自变量的取值。当函数值变化比较缓慢时，自变量的取值间隔大一点；当函数数值变化比较剧烈时，自变量的取值会取得小一点，这样，就可以充分方便地保证绘图的质量和效率。

在MATLAB中，fplot命令的常用调用格式如下。

- fplot(function,limits,tolerance,linespec)：在这个命令中，参数function表示需要绘制曲线的函数名称；limits表示绘制图形的坐标轴取值范围，可以有两种方式：[xmin xmax]表示横坐标轴的取值范围，[xmin xmax ymin ymax]则表示x、y坐标轴的取值范围；tol表示函数相对误差容限，默认值为2e-3；linespec表示线型的设置，默认值为数据点连线。
- [...] = fplot(function,limits,tol,linespec,P1,P2,...)：在这个命令中，fplot函数通过给定参数计算取值后，向函数传递P1,P2,...等函数参数数值。

6.2.14 使用fplot命令绘制图形

例6.14 在MATLAB中绘制函数$y_1 = 200\frac{\sin x}{x}$和$y_2 = x^2$的图形。

提示

[illegible]

step 1 打开 MATLAB 的 M 文件编辑器，在其中输入下面的函数表达式

```
function Y = fplot_fun(x)
Y(:,1) = 200*sin(x(:))./x(:);
Y(:,2) = x(:).^2;
```

[illegible]

step 2 在 MATLAB 的命令窗口中输入下面的语句

```
>> fh = @fplot_fun;
>> [X,Y]=fplot(fh,[-20 20],2e-4);
>> L=size(X);
>> px=-20:40/(L(1)+1):20;
>> py=fplot(fh,px);
>> subplot(2,1,1),plot(X,Y,'Linewidth',2),title('fplot')
>> [illegible]
```

step 3 语句执行结束，输入任意字符，按"Enter"键，程序的绘图结果如图所示。

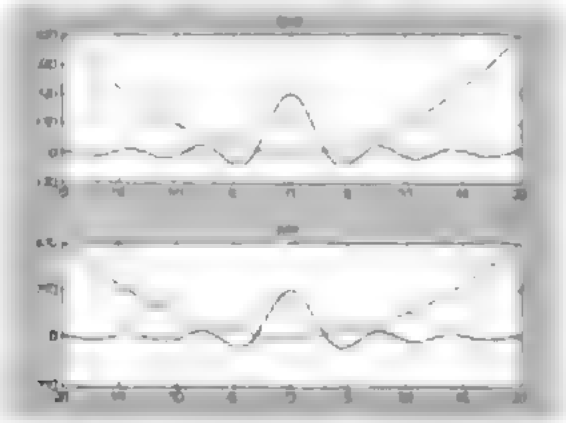


图 6.20 使用 fplot 命令绘制图形

说明

在上面的步骤中，我们使用 fplot 命令分别画了函数的图形，然后定义自变量的取值范围，再使用 fplot 命令绘制函数的图形。虽然看起来是完全一样的，但是在处理速度上是有很大区别的。

6.2.15 ezplot 命令

ezplot 命令是 MATLAB 中的一个绘图命令，它是 fplot 命令，其命令名称是前两个字符"ez"的英文缩写"Easy to"，表示"容易的"命令。这个命令的输入参数，不需要用户创建图形所需的数据，只需要输入字符串函数或者符号函数的表达式。

在 MATLAB 中，ezplot 命令的常用调用格式如下

- ezplot(fun)：在默认的自变量范围 $[-2\pi,2\pi]$ 内绘制函数 fun 的曲线。
- ezplot(fun,[min,max])：在用户自定义的自变量范围 [min,max] 内绘制 fun 的曲线。
- ezplot(fun,[min,max],fig)：在制定的图形窗口 fig 中，在自定义的自变量范围内，绘制函数 fun 的曲线。

在上面的函数中，参数可以是字符表达式，符号函数，内联函数等。但是如果的函数参数只是单一的函数，在默认情况下，ezplot命令会用表达式和自变量写出函数名和自变量的名称。为了防止混淆，建议用户以明确指定参数的ezplot命令格式来指定函数名称。

6.2.16 使用ezplot命令绘制图形

例6.15 在MATLAB中绘制函数 $y=\begin{cases} e^{-4t}\sin(t+2t) \\ \end{cases}$ 的图形。

step 1 在MATLAB的命令窗口中输入下面的内容

```
>> syms t
>>y= 4*exp(-2*t)*sin(t+2*t);
>>ezplot(y,[ pi,3*pi ]);grid
```

step 2 在命令行结束，输入分号之后，按"enter"键，得到该曲线函数的图形。

图6.21 使用ezplot命令绘制图形

从上面的图形中可以看出，首先使用syms命令将t设置为符号变量，然后输入符号表达式，最后使用ezplot命令来绘制图形。用户并没有为函数提供任何的自变量的数据，MATLAB自动为其设定自变量范围，同时计算相应的函数数值。

在ezplot命令中，不像在前面章节中plot命令中的函数作图语句，函数名、自变量、函数表达式、时间、一个图形窗口中绘制多个函数图形，在前面章节中介绍的grid、zoom等命令也是适用于ezplot命令的。

6.3 绘制三维曲线

从本节开始，将详细介绍在MATLAB中绘制三维曲线的命令和方法。与前二维绘图一样，三维绘图在仿真和工程应用中，但是三维曲线在很多方面是二维图形曲线没有涉及的。本节将详细介绍三维曲线的命令方法。

6.3.1 plot3 命令

和二维曲线命令plot类似，plot3命令是绘制三维曲线的基础命令，也是最简单的命令，其调用格式和plot非常相似。其具体的调用格式如下

[illegible] plot3([illegible]PropertyName,PropertyValue,[illegible]) 对于不同情况下的输入参数X、Y和Z，该命令会得出不同的曲线结果。

- [illegible]维向量，MATLAB会绘制以X、Y和Z[illegible]为x、y、z坐标的一维曲线。
- [illegible]，MATLAB会绘制以X、Y和Z[illegible]为x、y、z坐标的三维曲线。曲线的个数等于矩阵的列数。
- [illegible]，参数[illegible]，曲线的线型、颜色和数据点标等。参数PropertyName是曲线对象的属性名，PropertyValue是对应属性的取值。

6.3.2 plot3 命令实例

例6.16 在MATLAB中绘制圆锥螺线的三维图形。

根据高等数学知识，圆锥螺线的三维参数方程如下

$$\begin{cases} x = v\sin\alpha\cos wt \\ y = v\sin\alpha\sin wt \\ z = v\cos\alpha \end{cases}$$

[illegible]，圆锥角[illegible]，旋转角速度[illegible]，旋转半径[illegible]，[illegible]。在本实例中仅保留参数t来绘制三维曲线，具体步骤如下

Step 1 在MATLAB命令窗[illegible]输入下表的语句

```
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
>>grid on
```

Step 2 查看绘图结果，输入[illegible]，按“Enter”键，得到圆锥曲线如图6.22所示。

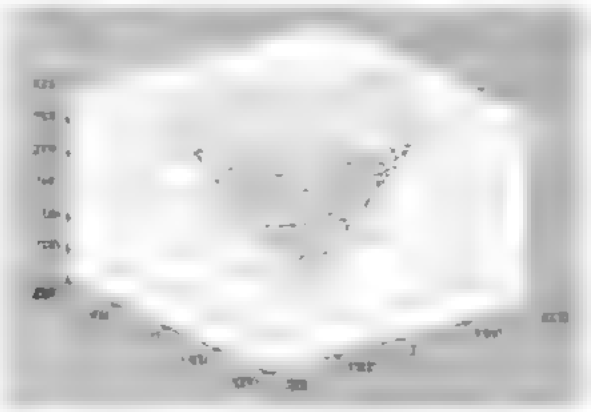


图6.22 绘制三维圆锥曲线

在 MATLAB 中，[illegible]命令plot3，[illegible]绘制[illegible]的一维曲线；对于有多个参数的曲线，需要使用其他的绘图命令。

6.3.3 曲线图命令

当绘制一维曲线的时候，除了需要绘制单根曲线，有时还需要绘制一组曲线的曲线簇和曲面图。为此，MATLAB提供mesh和surf命令，分别用来绘制三维曲线簇和曲面图。

在MATLAB中，绘制一维函数 $z=f(x,y)$ 时，用户需要分别准备自变量 x 和 y 的取值数据，然后通过下面的命令产生数据矩阵：

```
>> x=x1:dx:x2;y=(y1:dy:y2)';
>> X=ones(size(y))*x;
>> Y=y*ones(size(x));
```

在上述命令行语句中，[illegible]数据矩阵 x 、y 。[illegible]了一个[illegible]矩阵。其中，dx、dy表示两个自变量的步长间隔，而x、y使用冒号的命令产生数组，[illegible]。MATLAB会以数据矩阵为自变量取值，绘制相应的网格线。

在MATLAB中，mesh的常用调用方式如下：

- [illegible]为[illegible]、y轴向量，绘制网格线
- mesh(x,y,z)：最常用的网格线调用格式。
- mesh(x,y,z,c)：[illegible]

上面调用格式中，输入参数[illegible]是一致的，[illegible]调用方式中各个参数的含义。其中，x、y[illegible]自变量"格点"矩阵，z[illegible]为"格点"[illegible]矩阵，[illegible]。[illegible]，默认值为[illegible]，[illegible]，MATLAB会[illegible]。

6.3.4 曲线图实例

例 6.17 在MATLAB中绘制函数Peaks的三维网格线。

Peaks是MATLAB的内置函数，[illegible]，它可以是可微（连续）函数的数值[illegible]。Peaks的参数x和y的取值范围是[-3,3]，函数peaks的结果如图一个N×N矩阵，通常被用来演示三维图形中的[illegible]；本书提供了[illegible]，可以绘制Peaks的曲线。

step 1 在MATLAB的命令窗口中输入下面的内容

```
>>z=peaks(25);
>>mesh(z)
>> colormap(cool)
```

step 2 查看[illegible]。输入代码之后，按"Enter"键，[illegible]。

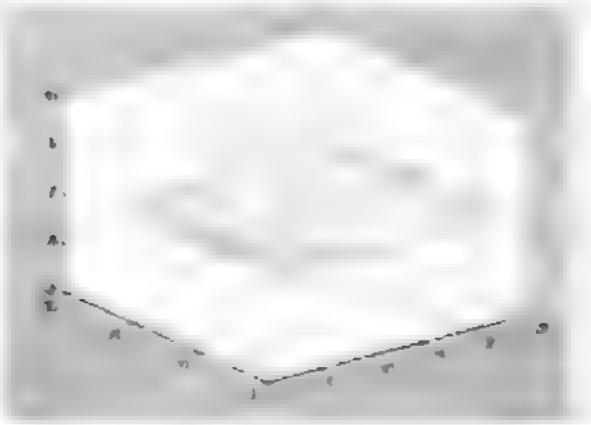


图 6.23 使用 mesh 命令绘制三维曲线

在上面的程序代码中，使用了 colormap 函数来为三维图形设置颜色。这个命令是 MATLAB 的内置函数，可以使用 help 和 doc 命令来对其进行解释。在命令窗口中选择“Help”→“Colormap”命令，打开“Colormap Editor”对话框，在该对话框中可以对三维图形进行更加精细的设置，如图 6.24 所示。

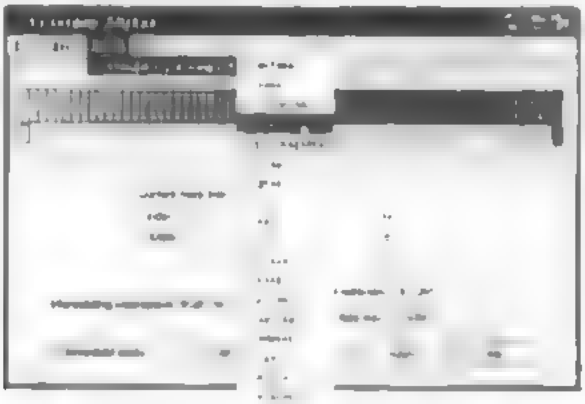


图 6.24 “Colormap Editor”对话框

在“Colormap Editor”对话框中，可以选择“Tools”→“Standard colormap”命令，查看 colormap 命令的第二参数，例如 hot、jet、bone 等，这些参数其实是 MATLAB 内部预定义的颜色参数，例如，cool 表示是“cyan-magenta”颜色，也可以来回移动光标查看。

colormap 函数可单独用来为三维曲线设置颜色，了解其语法格式与曲线的颜色设置十分方便，可以在 MATLAB 中查看关于该函数的帮助信息。

6.3.5 曲线图和等高线命令

在 MATLAB 中，还有很多和 mesh 命令一样可以绘制三维图形的命令，如 meshz、meshc 等，这些命令的调用格式都和 mesh 命令相同，只是在功能上有所不同，主要有以下几点。

- meshc 的功能是在 mesh 命令绘制的三维曲面上绘制等高线。
- meshz 的功能是在 mesh 命令基础上增加绘制边界的功能。

例 6.18 为 MATLAB 的绘制函数 meshc 的三维曲面的等高线和边界曲线。

step 1 在MATLAB的命令窗口中输入下面的内容

```
>> z=peaks(25);
>>meshc(z)
>>colormap(hsv)
```

step 2 查看效果，输入代码后，按“Enter”键，[illegible]

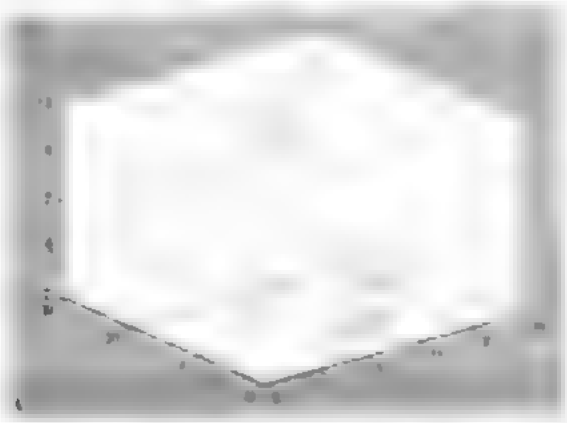


图6.25 绘制三维曲线的等高线

step 3 在MATLAB的命令窗口中输入下面的内容

```
>> clf
>> z=peaks(25);
>> meshz(z)
>> colormap(jet)
```

step 4 查看效果，输入代码后，按“Enter”键，[illegible]

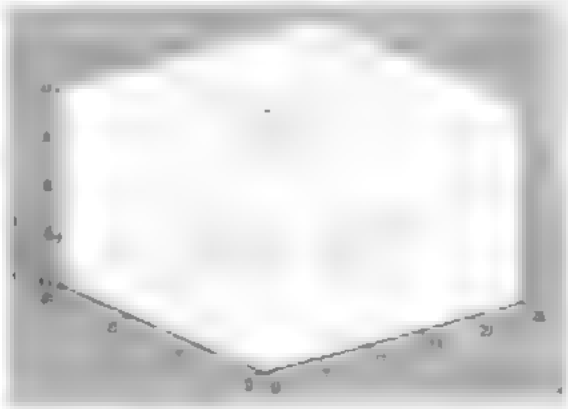


图6.26 绘制三维曲线的边界曲线

6.3.6 曲面图命令

在MATLAB中，绘制三维曲面图的主要命令是surf，该命令可以绘制出经着色的三维曲面。默认的着色方式是由全彩的方格网格[illegible]，每一个网格代表数据点的格网中表示的[illegible]值决定该网格的颜色。

在MATLAB中，surf命令的常见调用格式如下：

◆ surf(z)：以z矩阵的[illegible]，绘制曲面图。

- surf(x,y,z) 最常用的曲面绘线调用格式
- [illegible]

[illegible]

6.3.7 曲面图实例

例 6.19 在 MATLAB 中绘制函数 Peaks 的三维曲面图。

step 1 [illegible]

```
[illegible]
[illegible]
>>colormap(hsv)
```

step 2 [illegible]

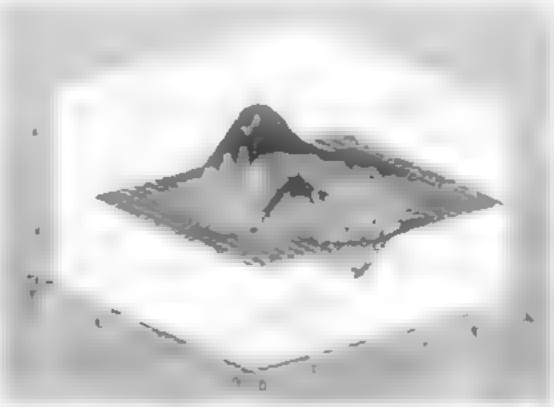


图 6.27 使用 surf 命令绘制三维曲面图

[illegible] 这些命令主要功能如下

- [illegible] 得到的。
- surfc 的功能是在 surf 命令的作用之上增加曲面的等高线。

例 6.20 [illegible]

step 1 [illegible]

```
>> z=peaks(25);
[illegible]
>>shading interp
>>colormap copper
```

step 2 [illegible]

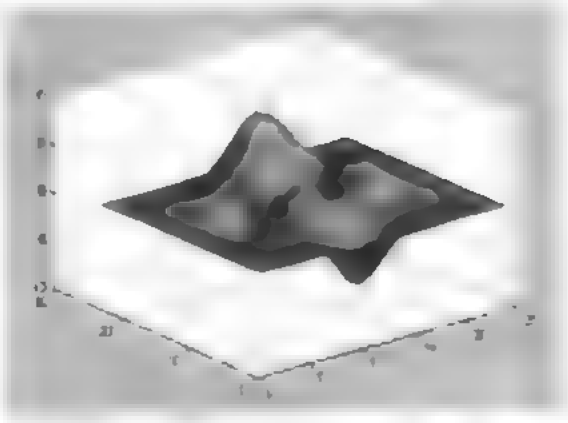


图 6.28 绘制明影效果的曲面

step 3 在 MATLAB 的命令窗口中输入下面字符串

```
>> z=peaks(25);
>> surfc(z)
>> colormap hsv
```

step 4 在编辑一结束，输入代码之后，按“Enter”键，观察绘图形如图 6.29 所示。

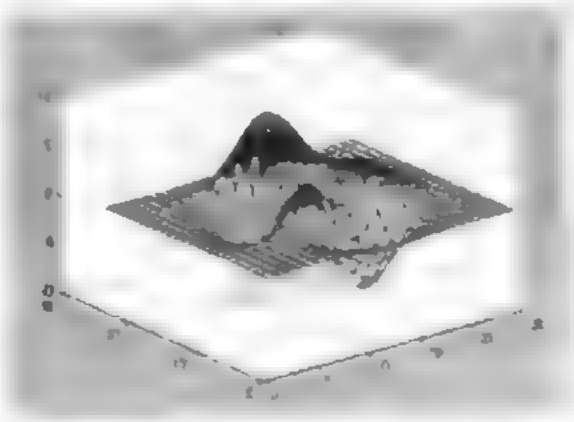


图 6.29 绘制三维曲面的等高线

6.4 特殊图形

在 MATLAB 中，为用户提供了不少特殊图形的命令，使用这些绘图命令可以快捷地绘制出一些特殊图形，如直方图、饼图、矢量图等。这些图形使用基础的 plot 命令也是可以绘制的，但是操作步骤比较复杂，而使用 MATLAB 内部已经设置好的命令则可以使绘制特殊图形变得简单。

本节将介绍 MATLAB 中一些典型的二维和三维特殊图形的命令，你将在 MATLAB 中绘制出示例图形。这些命令有的是二维和三维通用的，有的只适用于二维曲线，有的只适用于三维曲线。

6.4.1 area 命令

在 MATLAB 中，为用户提供了绘制面积图的命令 area，该命令通过填充绘制一维曲线图。这个命令的主要特点是，在图形中绘制多条曲线时，将各曲线相邻区域的曲线下填充，以反映各曲线

area 命令的常见调用格式如下：

- area(Y)：这是这种形式中最简单的命令，输入向量Y的值作为横坐标轴，以向量Y的元素数值为纵坐标轴来绘制面积图。
- area(X,Y)：当X和Y都是向量的时候，这个命令绘制的图形和plot(X,Y)相同，只是在默认的X轴第一值和数值Y之间有填充效果。当X是向量，Y是矩阵时候，以向量X为横坐标轴，绘制多条以一维字母数值为纵坐标数值的填充图形。
- area(...,'LineWidth',...)：在这个命令中，用户可以设置填充区域的数值，如果没有指定该参数的数值，MATLAB会将该数值设置为0。

对于area命令绘制的基础图形，前面介绍的所有通用命令都可以使用，例如grid、colormap等命令都可以使用在该命令绘制的图形中。

6.4.2 面积图实例

例6.21 在MATLAB中绘制面积图。

step 1 在MATLAB中的命令窗口中输入下面的命令。

```
>> Y = [ 2, 4, 5;
3, 1, 9;
5, 3, 5;
2, 6, 1];
>>area(Y,'LineWidth',2)
>>grid on
>>colormap summer
>>set(gca,'Layer','top')
```

step 2 稍等片刻，输入回车后，按"Enter"键，绘制的面积图形即开。

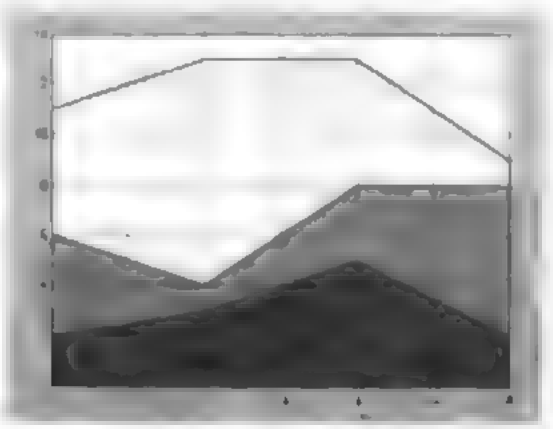


图6.30 完成的面积图形

注意：在这段程序的最后，使用了set(gca,'Layer','top')语句来设置坐标轴的属性，设置为最顶层，这样坐标轴的网格线就位于图形之上。

6.4.3 bar命令

在MATLAB中，提供命令bar和bar3来绘制二维和三维垂直条形图，提供命令barh和bar3h来绘

[illegible]

在 MATLAB 中，命令 bar 的常用调用方式如下

- bar(Y)　为向量 Y 中的每一个元素绘制一个条形。
- [illegible]
- [illegible] 参数 width 决定条形宽度，[illegible] 默认值是 0.8，若设置参数 width 设置为 1，则条形之间没有间隙。
- [illegible] 参数 [illegible] 'group' 和 'stack'，MATLAB 默认值是 'group'，[illegible] 'stack'，[illegible] MATLAB 的[illegible]。

命令 bar 和命令 plot 类似，[illegible] bar 命令绘制的是二维直方图，而且在命令 bar(...) [illegible] 参数，[illegible] MATLAB 中参数的值是 'detached'。[illegible]

[illegible]

6.4.4 直方图实例

例 6.22 [illegible] 在 MATLAB 中，[illegible] 绘制多种直方图。

Step 1 在 MATLAB 中的命令窗口中输入下面的命令

```
[illegible]
[illegible]
>> subplot(2,2,2); bar(income,'stack');title('Stack');
[illegible]
[illegible]
```

Step 2 [illegible]

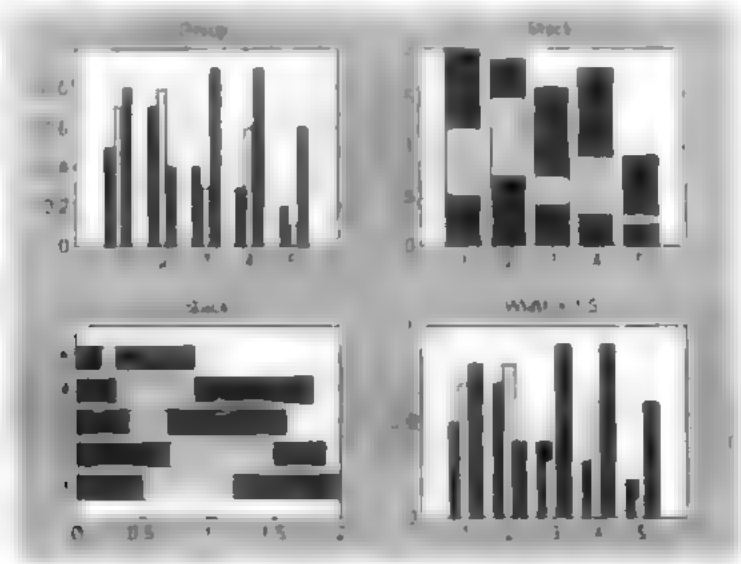


图 6.31　完成的直方图

[illegible]

例6.23 [illegible]

Step 1 在MATLAB中的命令窗口中输入下面的命令

```
>> Y=cool(8);
>> subplot(2,2,1);bar3(Y,'detached');title('Detached')
>> subplot(2,2,2);bar3(Y,'grouped');title('Grouped')
>> subplot(2,2,3);bar3(Y,'stacked');title('Stacked')
>> [illegible]
```

Step 2 [illegible]

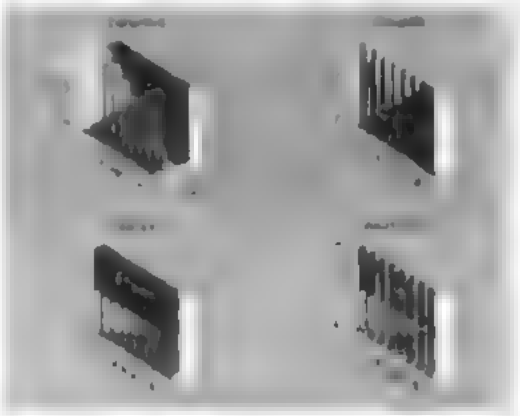


图6.32 [illegible]

6.4.5 pie命令

[illegible]

在MATLAB中，pie的常见调用方式如下

- [illegible]
- [illegible]
- pie(...,labels) 参数labels用来定义对应扇形的标签。

[illegible]

6.4.6 二维饼图实例

例6.24 [illegible]

step 1 在MATLAB的命令窗口中输入上面的命令。

```
>> x = [1 2.5 1.8 3.6 2.4];
>>subplot(2,2,1);pie(x,[1:5],{'Dep1','Dep2','Dep3','Dep4','Dep5'})
>>subplot(2,2,2);pie(x);colormap cool
>>subplot('position',[.2,[illegible]]);explode=[1 0 0 1 0];pie(x,
explode)
```

step 2 查看图形结果，输入代码之后，按"Enter"键，即可在弹出窗口中显示图形。

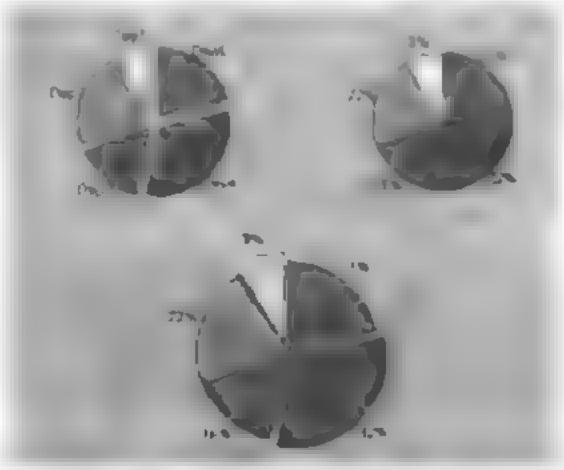


图6.33 绘制二维饼图

6.4.7 三维饼图实例

例6.25 在MATLAB中绘制一维饼图，以分析各个部门销售所占的比例。

step 1 在MATLAB的命令窗口中输入下面的命令。

```
>> x = [1.2 1.8 3.6 2.4];
>>subplot(2,2,1);pie3(x,[1:5],{'Dep1','Dep2','Dep3','Dep4','Dep5'})
>>subplot(2,2,2);pie3(x);colormap cool
>>subplot('position',[.2,[illegible]]);explode=[1 0 0 1 0];pie3(x,
explode)
```

step 2 查看图形结果，输入代码之后，按"Enter"键，即可在弹出窗口中显示图形。

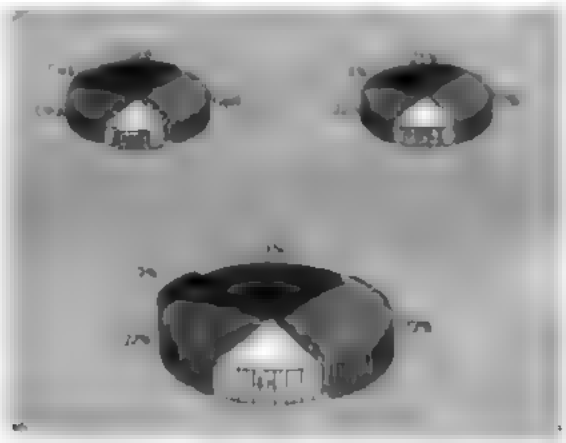


图6.34 绘制三维饼图

6.4.8 quiver 命令

矢量场在工程领域有着广泛应用，例如在数理或者通信系统中有着广泛应用。在MATLAB中，可以使用quiver命令来绘制矢量场图形，

quiver命令的常用调用格式如下

- quiver(x,y,u,v)：在坐标为x,y的点上使用箭头表示绘制矢量，其中u,v是对应坐标点的速度分量，参数x，y，u，v必须是同维向量
- quiver(u,v)：在x，y平面坐标系中等间隔的坐标点上绘制矢量u,v
- quiver(...,scale)：参数scale是用来控制绘制矢量长度的参数，其默认数值是1，可以根据需要调整scale的数值，以使绘制的矢量清晰可见。

6.4.9 矢量图实例

例 6.26 在 MATLAB 中绘制 peaks 函数得到的矢量图。

step 1 在MATLAB的命令窗口中输入下面的命令。

```
>> n = -2.0:.2:2.0;
>> [X,Y,Z] = peaks(n);
>> [U,V] = gradient(Z,.2);
>> quiver(X,Y,U,V)
>> hold on
>> contour(X,Y,Z,10)
```

step 2 查看绘图结果，输入代码完成后，按"Enter"键，得到的曲线如图6.35所示。

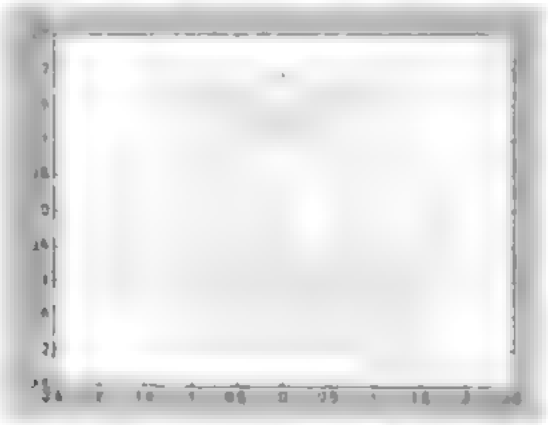


图 6.35 绘制矢量图

在上面的程序中，首先使用函数gradient产生peaks函数数据的梯度数值，然后将其结果作为quiver的输入数值，产生矢量图的数据。在上面程序的最后一行，使用contour函数绘制等高线，使整个图形更加美观。

6.4.10 contour 命令

等高线原来是属于地理学的概念，它表示海拔、气压等高度或者深度。在MATLAB中，等高线是通过给定函数的一个"水平面"，等高线把曲面上高度相同的点连在一起。在默认情况下，MATLAB就是画出相当于一系列海拔高度z值的等高线。在MATLAB中，提供了contour和contour3

[illegible]

其中，contour 命令的常用调用方式如下

- contour(Z)　变量 Z 就是需要绘制等高线的函数表达式
- contour(Z,n)　参数 n 是所绘图形等高线的参数
- [illegible]
- [C,h] = contour(...)　C 是等高线矩阵，h 是等高线的句柄。

6.4.11 二维等高线实例

例 6.27　[illegible]

step 1　[illegible]

```
>> Z = peaks;
>> [C,h] = contour(interp2(Z,4));
>> text_handle = clabel(C,h);
>> set(text_handle,'BackgroundColor',[1 1 .6],...
    'Edgecolor',[.7 .7 .7])
>> grid on
```

step 2　[illegible]

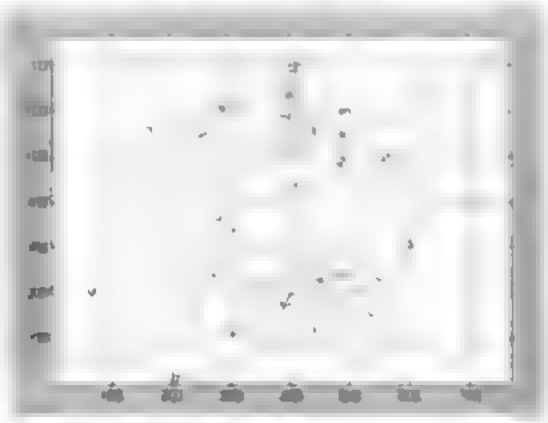


图 6.36　绘制二维等高线

6.4.12 三维等高线实例

例 6.28　[illegible]

step 1　[illegible]

```
>> [X,Y] = meshgrid([-2:.25:2]);
>>Z = X.*exp(-X.^2-Y.^2);
[illegible]
>>surface(X,Y,Z,'EdgeColor',[.8 .8 .8],'FaceColor','none')
>>grid off
[illegible]
>>colormap hot
```

step 2 [illegible]

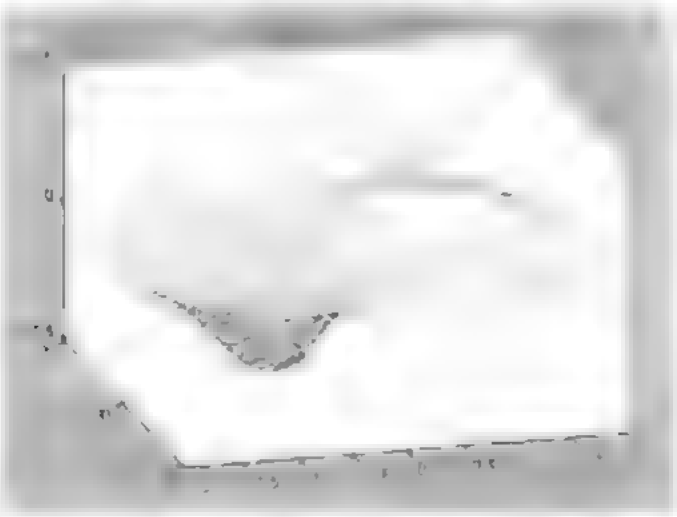


图 6.37 绘制三维等高线

[illegible]

6.4.13 伪色彩图

[illegible]

在 MATLAB 中，pcolor 命令的常见调用格式如下

- [illegible]
- [illegible]

例 6.29 在 MATLAB 中绘制函数 peaks 的伪色彩图。

step 1 在 MATLAB 命令窗口中输入下面命令

```
>> [illegible]
>> pcolor(x,y,z);
>> colormap hsv
>> shading interp
>> hold on; C=contour(x,y,z,4,'k') ;
>> clabel(C);
>> zmax=max(max(z));zmin=min(min(z));caxis([ zmin,zmax] );
>> colorbar;
```

step 2 [illegible]

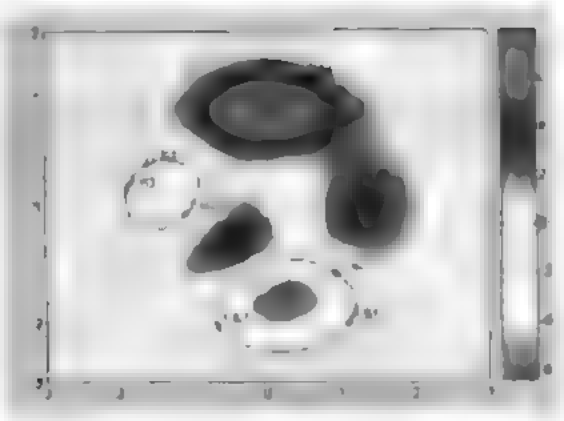


图 6.38 绘制伪色彩图

在上面的程序代码中，使用了 shading interp 命令设置曲面为平滑插值方式，然后使用前面介绍的 contour 命令来绘制等高线，并使用 clabel 命令为等高线添加数值标签。在程序执行后，为整个图形添加了颜色标尺，显示各个颜色对应的数值。

上面这些命令对图形的颜色属性的控制非常灵活，在后面章节中，将会对上面这些命令进行详细介绍。

6.4.14 errorbar 命令

误差线在数据分析中有着重要的应用，在绘制图形时添加数据误差线，可以方便用户直观地查看各个数据点的误差变化范围。在 MATLAB 中，为用户提供了 errorbar 命令，可以绘制一条带误差棒的曲线，其中误差棒反映数据的置信区间或数据的误差。与前个命令输入参数是矩阵时，由矩阵的数据列画出误差棒。

在 MATLAB 中，errorbar 命令的常见调用格式如下：

- errorbar(Y,E)：绘制向量 Y 的曲线，然后显示在向量 Y 的每个元素的误差棒。误差棒表示曲线 Y 上面和下面的距离，因此误差棒的长度是 2E。
- errorbar(X,Y,L,U)：X、Y 和 L、U 分别是同样大小的向量。如果 L、U 参数都是向量，MATLAB 会绘制出带宽度为 L、U 的误差棒；如果 L、U 参数都是标量，MATLAB 会绘制出带宽度为 L、U 的误差棒，并对称误差棒于曲线上；如果 L、U 参数都是矩阵，MATLAB 会绘制出带宽度为 L(i,j)、U(i,j)，并对称误差棒于曲面点。

6.4.15 误差棒形图实例

例 6.30 在 MATLAB 中绘制函数 $y=\frac{1}{(x-3)^2+1}+\frac{1}{(x-9)^2+4}+5$ 的图形，并绘制对应数据点的误差棒。

Step 1 在 MATLAB 命令窗口中输入下面的命令：

```
>> x=0:0.5:16;
>> y= 1 ./ ((x-3).^2 + 1) + 1 ./ ((x-9).^2 + 4) + 5;
>> E = std(y)*ones(size(x));
>> errorbar(x,y,E,'r','LineWidth',2);
>> grid on
>> box on
```

Step 2 查看绘图结果。输入代码之后，按“Enter”键，得到的曲线绘图如图所示。

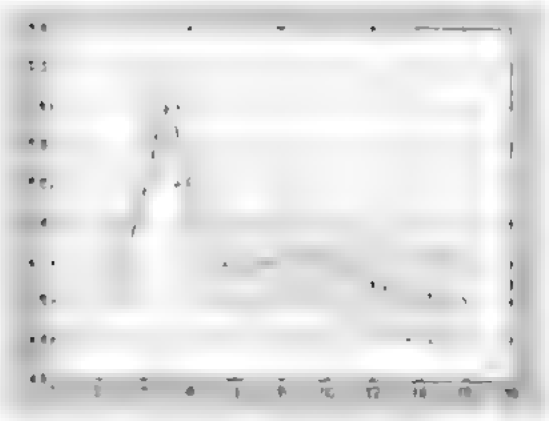


图6.39 绘制曲线的误差线

在上面的程序中，使用语句[illegible]，[illegible]函数为每个数据点绘制误差线。其中，[illegible]函数，要求[illegible]，[illegible]，将自变量x的[illegible]绘制，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]相等，[illegible]，曲线中每点的误差棒长度相等。

说明

从上面的程序语句中可以看出，误差棒的绘制需要[illegible]图形坐标进行绘制，例如添加[illegible]，设置坐标轴[illegible]，设置坐标轴的属性等。

6.4.16 stem命令

离散杆图也是一种最为常见的图形类型，这种图形是以坐标轴将坐标点用直线和基准线相连，也就相当于以数据点在坐标轴的垂线，数据点由数据标记来显示。在MATLAB中，提供二维和三维离散杆图的绘图命令stem和stem3。

在MATLAB中，绘图命令stem的常见调用格式如下

- [illegible]：绘制[illegible]维离散杆图，其[illegible]数据[illegible]，默认情况下以[illegible]坐标轴等距的、系统自动产生的数值序列
- [illegible]：参数x和y[illegible]，[illegible]，绘制以x为横坐标，y为纵坐标的离散杆图
- [illegible]：其[illegible]参数和前面的[illegible]，其中[illegible]设置直线的属性。

说明

关于命令stem3，其参数和stem大致相同，不同绘制离散杆图的参数为三维参数，其他参数都是相同的，读者可以查看MATLAB的帮助文件。

6.4.17 二维离散杆图

例6.31 在MATLAB中绘制函数$y=e^{-t/3}+\sin(1+2t)$的离散杆图，并绘制该函数的真实图形。

step 1 在MATLAB命令窗口中输入下面的命令

```
>> t=[0:0.5:25]';
>> y=exp(-t/3)+sin(1+2*t);
>> h=stem(t,y,'[illegible]','[illegible]');
>>set(get(h,'BaseLine'),'LineStyle',':');
```

```
>>stem(t,y,'MarkerFaceColor','r','LineWidth',1);
>>hold on
>>t2=[0:0.1:25]';
>>y2=exp(-t2/3)+sin(1+2*t2);
>>plot(t2,y2,'r','LineWidth',2);
>> box on
>> set(gca,'[illegible]','[illegible]')
```

step 2 按回车键，输入代码之后，按“Enter”键，得到如图6.40所示的结果。

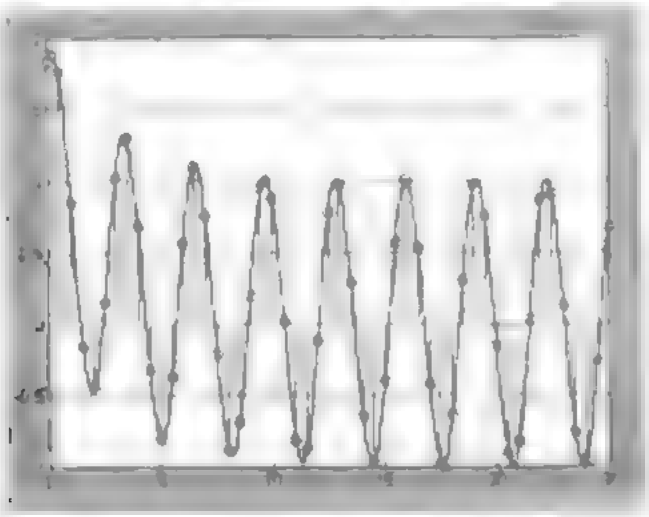


图6.40 绘制二维离散杆图

说明 从图中可以看出，离散杆图中的曲线连接了各个数据点的数值大小情况，离散杆图中的曲线连接了各个数据点的数值。

6.4.18 三维离散杆图

例6.32 在MATLAB中绘制函数 $y=e^{-at}$ 的三维离散杆图。

step 1 在MATLAB命令窗口中输入下面的命令

```
>> t = 0:0.1:10;
>>a = 0.1+i;
>> y = exp(-a*t);
>> stem3(real(y),imag(y),t)
>>hold on
>>plot3(real(y),imag(y),t,'r')
>>hold off
>>xlabel('real(y)')
```

step 2 查看语句结果，输入代码之后，按“Enter”键，得到如图6.41所示的结果。

6.4.19 散点图

在MATLAB中，提供了scatter、scatter3和scatterhist三种命令来绘制散点图，这三种不同命令的绘图效果各有特色。其中，scatter命令的第一种格式为：

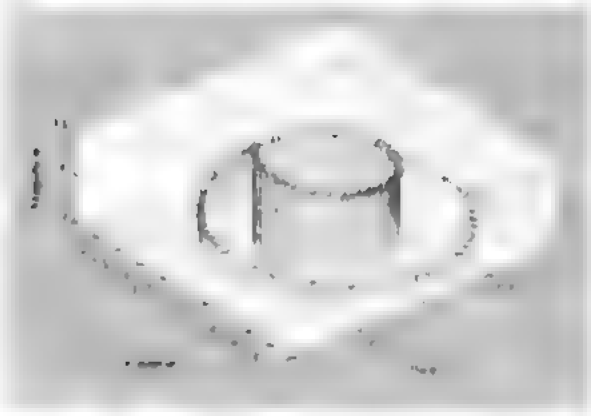


图6.41 绘制三维离散柱图

- [illegible]，绘制出[illegible]，[illegible]，因此MATLAB就采用默认的颜色和大小绘制数据点。
- scatter(X,Y,S) 参数S表示的是绘制数据点的颜色和大小。

[illegible]
MATLAB [illegible]

例6.33 在MATLAB中绘制柱体的三维散点图。

step 1 在MATLAB命令窗口中输入下面的命令

```
>> [x,y,z] = cylinder(20);
>>X = [x(:)*.5 x(:)*.75 x(:)];
[illegible]
[illegible]
>>S = repmat([1.5 2.5 3.5]*25,prod(size(x)),1);
>>C = repmat([5 6 7],prod(size(x)),1);
>>scatter3(X(:,1),X(:,2),X(:,3),S(:,1),C(:,1),'filled' ,'MarkerEdgeColor',[0]
```

step 2 查看命令结果，输入代码之后，按"Enter"键，得到的柱体散点图如图6.42所示。

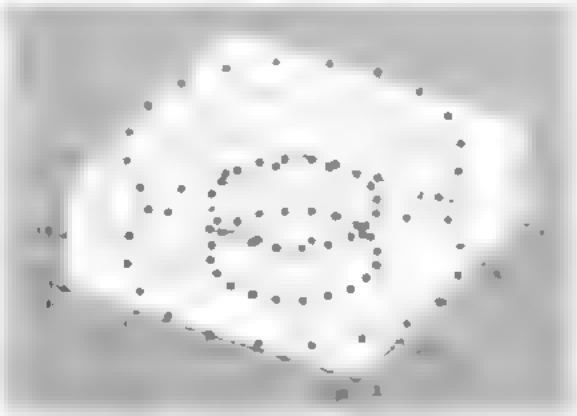


图6.42 柱形的三维散点图

[illegible]

plotmatrix命令的常见格式如下

plotmatrix(X, LineSpec)，其中X和Y分别是多维数组，X的维度是$p \times n$，Y的维度是$p \times m$。命令plotmatrix将会绘制一个分割成$m \times n$个子图的图表，其中第i、j个子图的内容是由Y第i列和X第j列数据绘制而成的。

例6.34 在MATLAB中使用plotmatrix命令对数据之间的统计关系

step 1 在MATLAB命令窗口中输入下面的命令

```
>> X=randn(150,2);
>>Y=randn(150,2);
>> subplot(1,3,2),plotmatrix(X,X)
>>subplot(1,3,1),plotmatrix(X)
>>subplot(1,3,3),plotmatrix(X,Y)
```

step 2 查看绘制结果，输入完成之后，按"Enter"键，得到的绘制结果如图6.43所示。

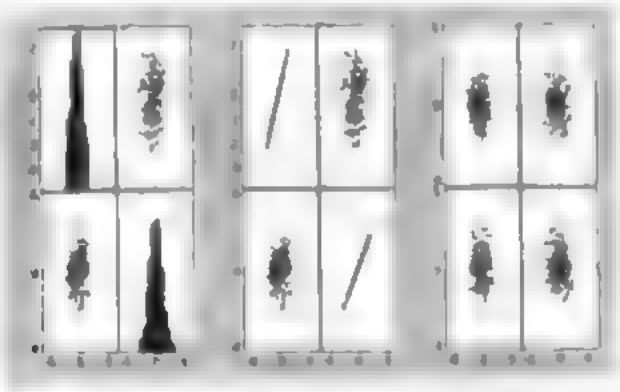


图6.43 表现数据的统计特征

从上面的图形中可以看出，每个的子图标中的直方图表示变量X数据列的数据大致服从钟形分布的正态分布，而其他散点图表示X的两个数据列之间是不相关的，中间的子图标表示X的两个数据列是正相关的，第三的子图标表示X、Y之间完全不相关。

6.4.20 极坐标图形

在前面小节中，读者已经了解在MATLAB直角坐标系中各种绘图命令和编辑命令。在MATLAB中，除了可以在熟悉的直角坐标系中绘图之外，还可以在极坐标或者柱坐标中绘制各种图形。

在MATLAB中，绘制极坐标图形的主要命令是polar，其常用的调用格式如下：

- polar(theta,rho)，该命令使用数组theta和数组rho绘制极坐标图形。
- polar(theta,rho,'linespec')，参数LineSpec表示极坐标图形中的线条线型、标记和颜色等主要属性。

例6.35 在MATLAB中绘制简单的极坐标图形。

step 1 在MATLAB的命令窗口中输入下面的命令

```
>> t = 0:.01:2*pi;
>> polar(t,sin(2*t).*cos(2*t),'--r')
```

step 2 查看图形结果。输入代码之后，按 Enter 键，得到如图 6.44 所示。

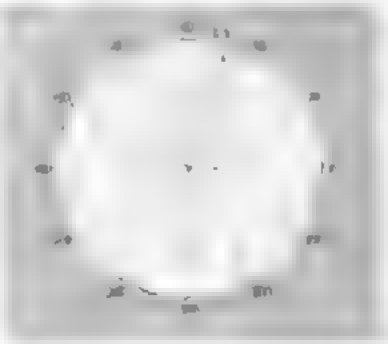


图 6.44 简易极坐标图形

6.4.21 柱坐标图形

在 MATLAB 中，绘制柱坐标图形的，要函数是 pol2cart。这个命令用于将极坐标或者柱坐标的数值转换成直角坐标系中的坐标值，然后使用一维绘图命令进行绘图，也就是在直角坐标系统下绘制使用柱坐标值描述的图形。

例 6.36 在 MATLAB 中绘制简单的柱坐标图形。

step 1 在 MATLAB 命令窗口中输入下列的命令。

```
>> theta=0:pi/50:4*pi; rho=sin(theta);
>> [t,r,z]=pol2cart(theta,rho,rho);
>> z=r.*t;
>> [X,Y,Z]=pol2cart(t,r,z);
>> mesh(X,Y,Z)
```

step 2 查看图形结果。输入代码之后，按 Enter 键，得到如图 6.45 所示。

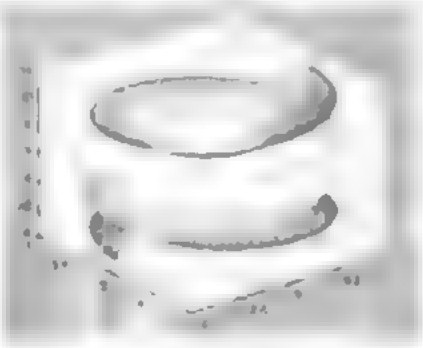


图 6.45 柱坐标图形

6.5 四维图形

对于二维图形，可以表示 z=f(x,y) 的函数关系来绘制图形。该函数关于变量 x 和 y 的变量的维度和，就是二维的。在实际生活和科学研究中，有时会遇到自变量个数大于两个，这个时候自变量的定义域是整个二维空间。与计算机有显示维度，只能表示一个空间变量，不能表示第四

[illegible]

6.5.1 slice 命令

[illegible]数上的四维表现。slice 命令的常用调用格式如下。

- [illegible]下点的切片图，各点的坐标由数量向量 sx,sy,sz 来指定。
- [illegible]
- [illegible]了一个曲面，同时会在曲面的点上计算超立体 v 的数值
- [illegible] nearest 等，分别对应不同的插值方法。

[illegible]

6.5.2 切片图实例

例 6.37 [illegible]

step 1 在 MATLAB 命令窗口中输入下面的命令

```
>> [x y z v] = flow;
>> x1=min(min(min(x)));x2=max(max(max(x)));
>> sx=linspace(x1+1.5,x2,4);
>> slice(x,y,z,v,sx,0,0);
>> view([-33 36]);
>> shading interp,
>> colormap hsv;
>> alpha('color')
>> colorbar
```

step 2 [illegible]

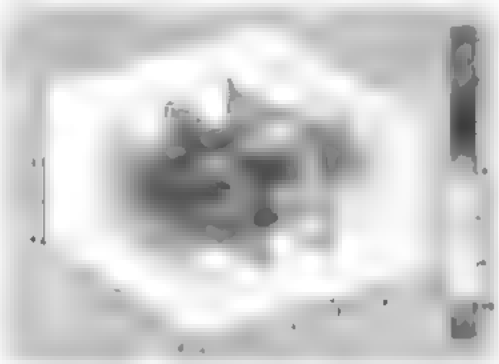


图 6.46 切片图

[illegible]

6.5.3 切面等位线图实例

例 6.38 [illegible]

step 1 [illegible]

```
>> [x y z v] = flow;
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
>>view([-12 30])
>>colormap cool
>>box on
[illegible]
```

step 2 [illegible]

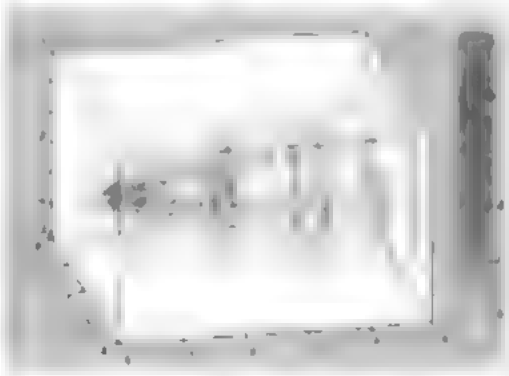


图 6.47 切面等位线图

6.5.4 流线切面图实例

例 6.39 在 MATLAB 中绘制函数 peaks 的流线切面图。

step 1 [illegible]

```
>> z = peaks;
[illegible]
>>shading interp
>>hold on
>> [c ch] = contour3(z,20); set(ch,'edgecolor','b')
>> [u v] = gradient(z);
>>h = streamslice(-u,-v);
[illegible]
>>colormap hsv
[illegible]
```

```
    zi = interp2(x,get(h(i),'xdata'),get(h(i),'ydata'));
    set(h(i),'zdata',zi);
end
>>view(30,50); axis tight; colorbar
```

Step 7 在命令窗口输入，输入上面的代码后，按"Enter"键，[illegible]

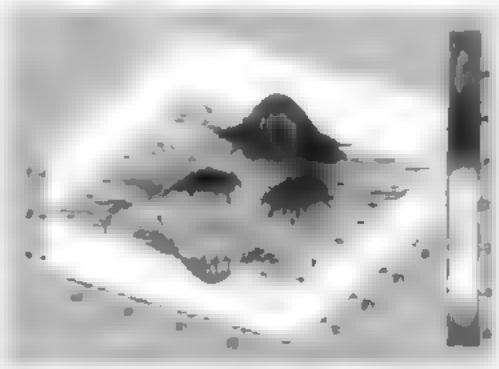


图6.48 流线切面图

6.6 三维图形的编辑

[illegible]

6.6.1 视角控制命令view

[illegible]

在MATLAB中，view命令的常见调用格式如下。

- view(az,el)、view([az,el])：[illegible]
- view(x,y,z)：[illegible]
- view(2)：设置默认的二维形式视点，其中az=0，el=90。
- view(3)：设置默认的三维形式视点，其中az=-37.5，el=30。

6.6.2 view命令实例

例6.40 在MATLAB中以不同的视角查看二维函数 $\begin{cases} x=10^{\sin 2t} \\ y=0 \end{cases}$ 的图形。

step 1 在 MATLAB 命令窗口中输入下面命令

```
>> t=0.01*pi:0.1*pi:3*pi;
[illegible]
[illegible]
[illegible]
view')
[illegible]
>>title('az Rotated to 52.5');view(-37.5+90,30)
[illegible]
to 10');view(-37.5,10)
[illegible]
```

step 2 回车后，在命令窗口中输入上面代码后，按"Enter"键，得到的图形如下所示。

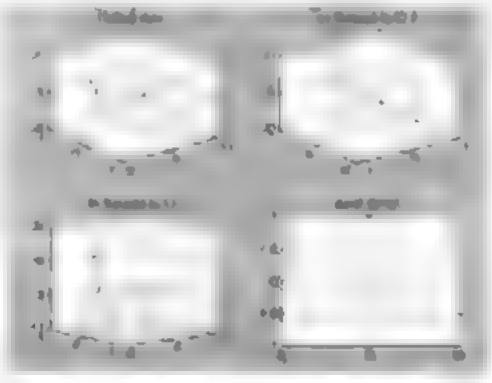


图 6.49 为三维图形设置视角

说明 在 MATLAB 中也可用向量来设置视角，使用命令 view(az,el)。该命令计算一个单位向量的方向，从原点开始，指向由方位角 az 和仰角 el 指定的观察点。方位角是在 x-y 平面中，关于原点的水平角度，单位为度。仰角是高于 x-y 平面的垂直角度，单位为度。

6.6.3 旋转控制命令 rotate

下面开始讲解旋转的视角命令 rotate。在 MATLAB 中，rotate 命令的常用调用格式如下：

- rotate(h,direction,alpha)：该命令将句柄 h 的对象绕方向旋转一个角度。其中参数 h 表示要被旋转的对象（例如线、面等）。参数 direction 有两种设置方法：direction 设置法，将其设置为[dx,dy,dz]，其单位是"数据"单位，直角坐标系，与坐标轴[x,y,z]。参数 alpha 是绕方向按照右手法则旋转的角度。

说明 和前面的命令 view 不同，如果使用命令 rotate，则将会旋转当前在图形中绘制的数据，而命令 view 只会改变图形的视角，不改变数据。

6.6.4 rotate 命令实例

例 6.41 在 MATLAB 中以不同的视角来查看函数 peaks 的三维图形。

step 1 [illegible]

```
>> z=peaks(25);
[illegible]
>>subplot(1,2,2);h=surf(z);title('Rotated')
[illegible]
>>colormap cool
```

step 2 [illegible] 按"Enter"键，[illegible]

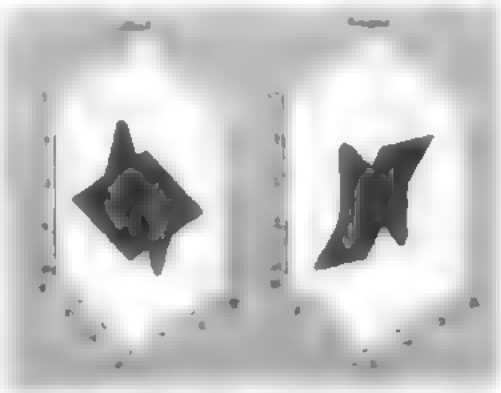


图 6.50 图形对象的旋转

说明 [illegible] 而是用命令rotate 旋转的是图形对象本身，而非[illegible]

在 MATLAB 中，[illegible] 函数来设定视角，[illegible] 由用户自定输入的视角参数。下面使用一个简单的实例来说明如何使用命令 rotate3d。

例 6.42 在 MATLAB 中动态调整函数 Peaks 的一维图形的视角。

step 1 [illegible]

```
>> surf(peaks(40));
```

step 2 [illegible] 按"Enter"键，[illegible]

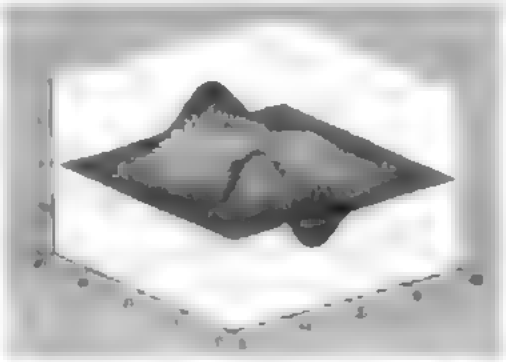


图 6.51 原始的 peaks 函数图形

Step 3 在 MATLAB 命令窗口中输入下面的命令

[illegible]

Step 4 [illegible]

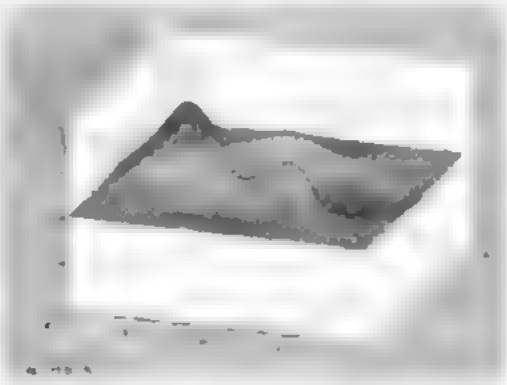


图 6.52 旋转三维图形

[illegible]

El=16。

[illegible]

6.6.5 设置背景颜色

[illegible]

在 MATLAB 中，[illegible]

- colordef white 将绘图的背景颜色设置为白色
- colordef black 将绘图的背景颜色设置为黑色
- [illegible]
- [illegible]

[illegible]

例 6.43 在 MATLAB 中为函数 peaks 的图形设置不同的背景颜色。

Step 1 在 MATLAB 命令窗口中输入下面的命令

```
>> subplot(2,2,1);colordef none;surf(peaks(35));Title('Default');
>> subplot(2,2,2);colordef black;surf(peaks(35));Title('Black');
>> subplot('position',[0.2,0.05,0.6,0.45]);
>> colordef white;surf(peaks(35));Title('White')
```

Step 2 查看显示结果。输入上述的代码后，按"Enter"键，在新的窗口中显示如图6.53所示。

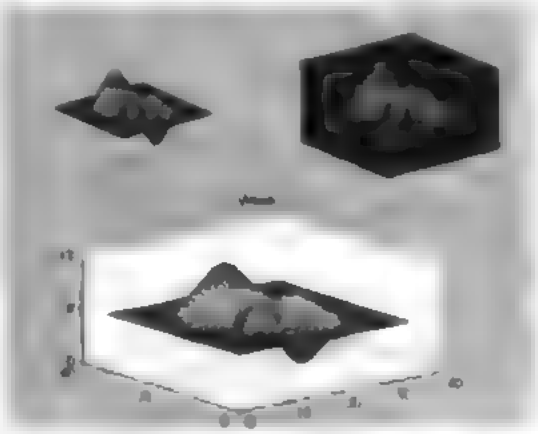


图6.53 设置不同的背景颜色

说明

在MATLAB中，除了可以设置图形的背景颜色之外，还可以设置图形坐标轴的颜色，实现这个功能的命令为whitebg。用于简洁，本书未涉及更开发的内容中，感兴趣的读者可以查看MATLAB的帮助文档。如果只需要改变坐标轴 whitebg命令无能为力，需要在图形窗口的属性中修改坐标轴颜色。

6.6.6 设置图形颜色

从上面小节的内容中可以看出，在MATLAB中可以很方便地设置图形的背景颜色。如果需要修饰图形的颜色，则还需要使用其他的命令。在MATLAB中管理图形颜色的重要命令是colormap，在前面小节中，读者已经多次接触到该命令，在本小节中，将详细介绍该命令使用方法以及相应的原理。

MATLAB采用颜色映像来处理图形颜色，也就是RGB色系。在MATLAB中，每种颜色都是由三个基色的数值表示的，数组元素R、G和B在[0,1]区间取值，分别表示颜色中红、绿、蓝三种基色的相对亮度，通过对R、G、B大小的设置，可以调制出不同的颜色。在MATLAB中，当使用绘图命令时，所有线条的颜色都是通过RGB调制出来的，表6.4列出一些常见的颜色配比方案。

表6.4 常见的颜色配比方案

基色			调制的颜色	对应的MALTAB符号
R	G	B		
0	0	1	蓝色（Blue）	b
0	1	0	绿色（Green）	g
1	0	0	红色（Red）	r
0	1	1	青色（Cyan）	c
1	0	1	品红色（Magenta）	m
1	1	0	黄色（Yellow）	y
0	0	0	黑色（Black）	k
1	1	1	白色（White）	w

[illegible]的颜色[illegible]，好[illegible]以使用MATLAB本身[illegible]、绘图命令[illegible]这些颜色，例如mesh、surf等。调用色图的基本命令是

```
colormap(h, map)
```

其中，函数的参数h是一个[illegible]，[illegible]数字图，这个句柄指定了调色的位图矩阵。在MATLAB中，每一个图形[illegible]都有一个句柄，在命令[illegible]通过句柄来定义，也可指定某个数据结构[illegible]。

MATLAB预定义[illegible]此色图矩阵的CM数值，[illegible]的维度由其调用格式来决定。

- CM：返回维度为64 × 3的色图矩阵。
- CM(m)：返回维度是m × 3的色图矩阵。

表6.5列出了MATLAB中的色图矩阵名称以及含义。

表6.5 MATLAB中的CM名称

名称	含义	名称	含义
[illegible]	红、黑色系	bone	蓝色调灰度色图
[illegible]	黑、[illegible]色系	copper	纯铜色调单色色图
gray	灰色调灰度色图	hot	黑红黄白色图
hsv	饱和色彩	jet	蓝头红尾的饱和色系

例6.44 在MATLAB中绘制函数peaks的图形，[illegible]函数图形的颜色。

step 1 在MATLAB命令窗口中输入下面的命令：

```
>> surf(peaks(100));
>> colormap(cool(64))
```

step 2 命令[illegible]结束，输入下面的命令，按"Enter"键，[illegible]。

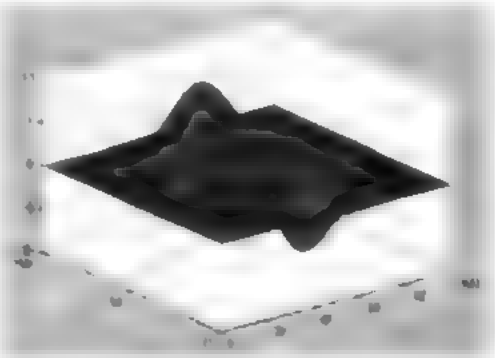


图6.54 使用cool绘图

6.6.7 设置数值轴的颜色

除了colormap函数外，MATLAB还提供多种颜色设置命令，[illegible]其他一些[illegible]颜色特性。其中，caxis命令和colormap命令是经常使用的命令，下面将详细介绍这两个命令的调用方式。

在MATLAB中，caxis命令的主要功能是设置数值轴的颜色，控制数值和颜色之间的对应关系。常用调用格式如下

- caxis([cmin cmax])：令[cmin cmax]范围外的色值相对应，并依此为其赋予颜色。

如果数据点的数值小于 cmin 或大于 cmax，则按照等于 cmin 或 cmax 来进行着色。

- caxis auto　MATLAB 自动计算出色值的范围。
- caxis manual　按照当前的色值范围来设置色彩范围。
- caxis(caxis)　和 caxis manual 实现相同的功能。

例 6.45　在 MATLAB 中绘制函数 peaks 的图形，同时设置该图形的颜色。

step 1　在 MATLAB 命令窗口中输入下面的命令。

```
>> z=peaks(45);
>> surf(z);
>> caxis([ -1.5 1.5])
```

step 2　查看图形结果。输入上面的代码后，按 "Enter" 键，得到如图 6.55 所示的图形。

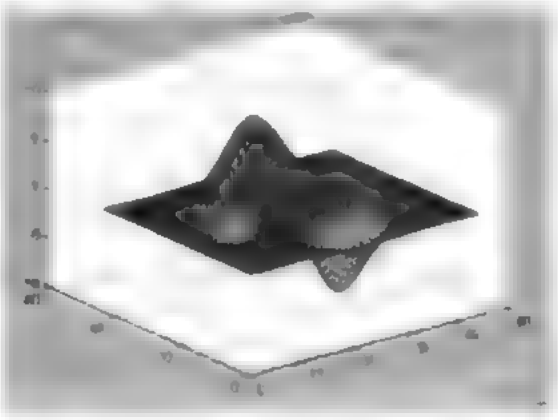


图 6.55　使用 caxis 命令为图形设置颜色

6.6.8 添加颜色标尺

在 MATLAB 中，colorbar 命令的主要功能是显示指示颜色刻度的颜色标尺，其常用语法格式如下。

- colorbar　更新最近生成的颜色标尺。如果当前坐标轴系统中没有颜色标尺，则在图形的右侧显示一个垂直的颜色标尺。
- colorbar('vert')　在新一个垂直的颜色标尺取当前的坐标轴系统中。
- colorbar('horiz')　添加一个水平的颜色标尺，它的坐标轴系统中。

例 6.46　在 MATLAB 中绘制函数 peaks 的图形，同时在图形中添加水平颜色标尺。

step 1　在 MATLAB 命令窗口中输入下面的命令。

```
>> z=peaks(45);
>> surf(z);
>> caxis([ -1.5 1.5])
>> colorbar('horiz')
```

step 2　查看图形结果。输入上面的代码后，按 "Enter" 键，得到的图形如图 6.56 所示。

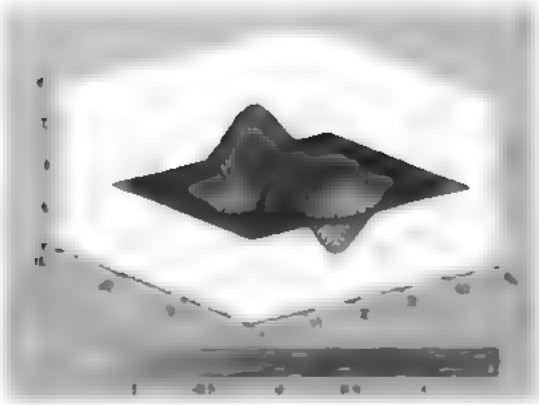


图6.56 为图形添加水平颜色标尺

说明

[illegible]

例6.47 在MATLAB中绘制函数[illegible]的图形，使用[illegible]设置不同的颜色

step 1 在MATLAB命令窗口中输入下面的命令

```
>>zmin=min(min(z));
>>zmax=max(max(z));
>>dz=zmax-zmin;
>>cm=[ hot;winter];
>>caxis([ zmin+dz*0.4,zmax])
>> subplot('position',[ 0.2,0.05,0.6,0.45]);surf(z);
caxis([ zmin,zmax+dz*0.8]);colorbar('vert')
```

step 2 查看[illegible]，输入下面命令[illegible]，按"Enter"键，得到[illegible]

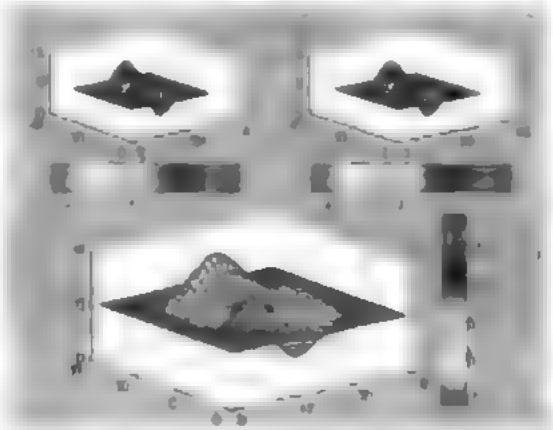


图6.57 为图形设置不同的颜色

说明

[illegible]

6.6.9 设置图形的着色

在 MATLAB 中，除[illegible]，还为用户提供了不同的着色方式，可以设置颜色的着色方式。对于绘图命令 mesh、surf、pcolor、[illegible]等创建了图形非数据处的着色，可用 shading 命令来设置。在 MATLAB 中，shading 命令有三种参数选项。

- shading flat：使用单一颜色来填充网格，网格线之间的线段，每段线段采用单一的颜色，这种方式的颜色取决于当前颜色图的颜色，或者该线段末端点所对应的颜色。
- shading interp：使用插值的方式实现着色。使用网格线连接，每段线段两端点的颜色由该线段两端点颜色决定，或者该线段末端点所对应的颜色。
- shading faceted：默认方式，使用单一颜色着色，但多边形内部颜色与 flat 方式相同，再在贴片的四周勾画黑色网线。

例 6.48 在 MATLAB 中绘制圆柱图形，然后使用一种不同的着色方式实现着色。

step 1 在 MATLAB 命令窗口中输入下面的命令。

```
>> t=0:pi/5:4*pi; [x,y,z]=cylinder(2*sin(t));
>> subplot(2,2,1);surf(x,y,z);shading interp;Title('interp')
>> subplot(2,2,2);surf(x,y,z);shading flat;Title('flat')
>> subplot(2,2,3);surf(x,y,z);shading faceted;Title('faceted')
>> colormap(hot)
```

step 2 查看绘图结果。输入上面的命令后，按"Enter"键，程序绘图结果如图 6.58 所示。

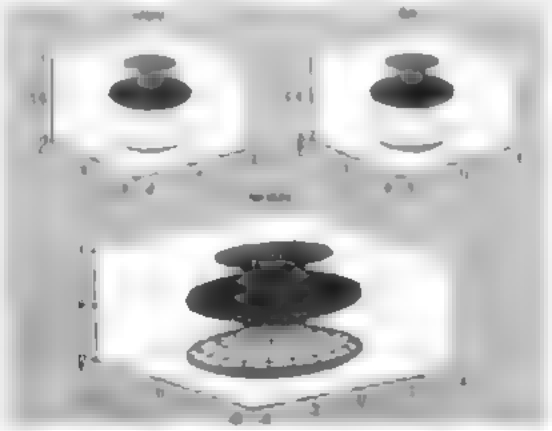


图 6.58 设置图形的不同的着色方式

说明 实际上，命令 shading 是编辑图形对象属性命令，[illegible]的 EdgeColor 和 FaceColor 属性，[illegible]命令得到的图形效果。

6.6.10 照明控制 light 命令

在三维图形中，除了填充颜色和着色处理之外，还需要设置光源的位置、照明模式和反射光处理，这样的图形才能显得更加美观。在本小节中，将介绍关于三维图形照明控制的相关命令，灵活使用这些命令可以使三维图形显得更加真实。

[illegible]，这是MATLAB[illegible]的基础命令，[illegible]，其调用格式如下

```
light('PropertyName',PropertyValue,...)
```

其中，[illegible]属性名称、[illegible]类型等的变量名，更详细的[illegible]请参阅相应的帮助文件。

关于light命令，需要了解的信息如下

- [illegible]，MATLAB[illegible]命令[illegible]，[illegible]命令，[illegible]相关属性被[illegible]
- [illegible]，[illegible]，MATLAB[illegible]默认设置的光源[illegible]光、无限远、通过[1，0，1]射向坐标轴
- [illegible]属性[illegible]数值'infinite'和'local'，前者表示光源位置是无限远，后者表示光照的位置是近光。

6.6.11 light命令实例

例6.49 在MATLAB[illegible]，[illegible]。

step 1 在MATLAB命令窗口中输入[illegible]命令

```
>> subplot(2,1,1);surf(peaks);light;title('Default')
>>subplot(2,1,2);surf(peaks);light('color','r','Position',[0 1 0],
'Style','local');
[illegible]
```

step 2 查看[illegible]，按"Enter"键，[illegible]

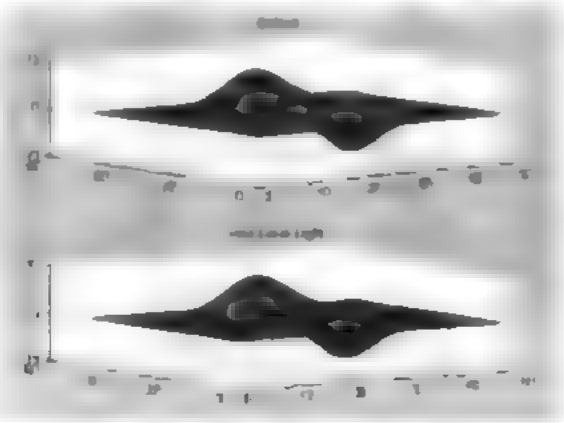


图6.59 不同的照明控制

在图6.59中，上面的"Default"子图[illegible]，下面的"Red-local light"子图[illegible]，[illegible]，[illegible]，[illegible]的图形中可以明显地看出光源的照亮效果的影响。

6.6.12 照明控制lighting命令

[illegible]，MATLAB[illegible]命令。使用该命

[illegible]的光照模式，但是lighting命令[illegible]在light命令[illegible]。该命令的调用格式如下。

- lighting flat　平面模式，这是系统默认模式，[illegible]对象的每个面一[illegible]，主要和faced一起配合使用。
- lighting gouraud　[illegible]模式。[illegible]顶点的颜色进行插补。
- lighting phong　[illegible]，[illegible]，其表现效果最好，但[illegible]。
- lighting none　关闭所有的光源。

6.6.13 lighting命令实例

例6.50　在MATLAB中绘制[illegible]，[illegible]效果。

step 1　在MATLAB命令窗口中输入下面的命令

```
>> t=0:pi/20:2*pi;
>> [x,y,z]=cylinder(1+cos(t));
>>subplot(2,2,1),surf(x,y,z),shading interp,lighting phong,title('Phong')
>>subplot(2,2,2),surf(x,y,z),shading interp,lighting flat,title
('Flat')
>>subplot(2,2,3),surf(x,y,z),shading interp,lighting gouraud,title
('Gouraud')
>>subplot(2,2,4),surf(x,y,z),light,shading interp,lighting none,title('None')
```

step 2　[illegible]，输入上面的代码，按"Enter"键，[illegible]。

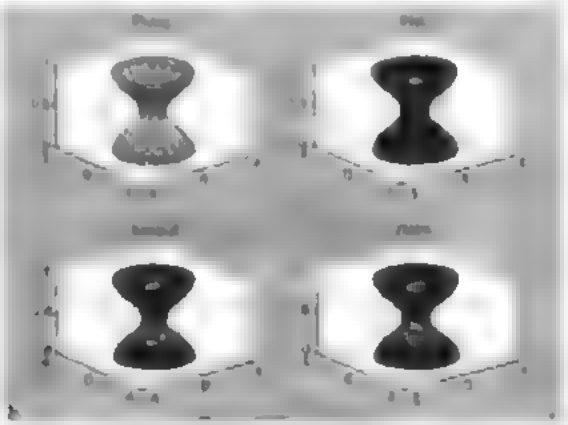


图6.60　不同的光照效果

6.6.14 材质控制material命令

material命令可以控制光照效果的材料属性，也就是说[illegible]表现出[illegible]反射方式。其语法的调用格式如下。

- material options　该命令使用预定义的反射模式，对于options的不同选项，其对应的选项含义如下

  - shiny　使对象比较亮[illegible]，镜面[illegible]，反射光[illegible]取决于光源颜色
  - dull　使对象比较暗淡，[illegible]，反射光[illegible]取决于光源颜色

- [illegible]：给对象不同部分赋予[illegible]，[illegible]表面光照效果，回到MATLAB内部的默认设置
- default：返回到MATLAB中的默认设置。

◆ material [illegible]命令使用[illegible]的[illegible]名称设置，[illegible]对材质参数进行自定义的设置，对应参数的含义如下：

- ka：设置无方向性、均匀的背景光的强度
- kd：设置无方向性的漫反射的强度
- ks：设置有镜反射光的强度。
- n：设置控制镜面亮点大小的镜面指数。
- sc：设置镜面颜色的反射系数。

6.6.15 material 命令实例

例6.51 在MATLAB中绘制peaks函数的三维图形，同时设置不同光照效果。

step 1 在MATLAB的命令窗口中输入下面的命令：

```
>> [x,y,z]=peaks(25);
[illegible]surf(x,y,z);shading interp,
[illegible]material [illegible]
>>light('color','r','Position',[0 1 0],'Style','local')
>>lighting phong
[illegible]surf(x,y,z);shading flat,
>>material shiny;
[illegible]
[illegible]
```

step 2 查看效果，输入上面的[illegible]，按'Enter'键，[illegible]，如下：

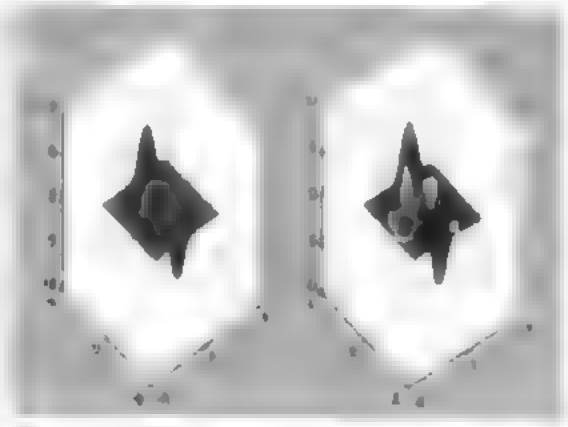


图6.61 设置不同的光照效果

6.6.16 透视控制

在MATLAB中，如果使用mesh、surf等命令绘制三维图形，在默认情况下，MATLAB会隐藏曲面的网格线。有时需要了解隐藏的网格线，这个时候用户需要使用透视控制命令。在MATLAB中，透视控制命令如下：

- hidden off　透视被缩压的图形
- hidden on　消除被缩压的图形

例 6.52　在 MATLAB 中演示透视效果。

step 1 在 MATLAB 命令窗口中输入下面的命令

```
>> [x,y,z]=sphere(50);[x0,y0,z0]=sphere(40);
>>surf(x0,y0,z0);shading,hold interp
>>hold on,mesh(x,y,z),colormap(hsv),hold off
>>hidden off
>> axis equal
>>axis off
```

step 2 查看绘制结果，输入上面的命令后，按“Enter”键，程序运行结束，绘制的图形如图 6.62 所示。

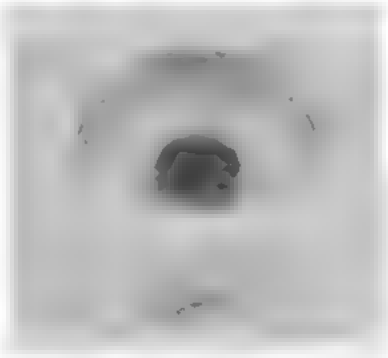


图 6.62　透视球体

6.6.17 透明控制原理

从 MATLAB 7.x 版本开始，系统增加了透明（transparency）处理的相关命令，该命令的主要功能是使不重要的或不希望显示部分图形的内部结构，相邻重叠节点的信息和光平滑和类似，透明技术可以为数据显示提供可视化的手段。

在 MATLAB 中，将透明度设置在[0，1]之间的数值，其中 0 表示完全不透明的，1 表示图形是不透明的，这种量化的数值被称为 Alpha 值。在 MATLAB 中，每个图元变量（曲面）都有一个透明表。在默认情况下，图元透明表是一个数组，其元素的数值都在[0，1]之间取值，其中第一个元素数值为 0，最后一个元素的数值为 1，其他元素按照线性递增方式进行排列。

限于篇幅，本节对于图形透明度管理将做下详细介绍。在本小节中，主要介绍 MATLAB 中透明度的处理方式，必须根据 $m \times n$ 的值矩阵 X、Y 和 Z 的坐标和得到的曲面对象，MATLAB 有如下三种透明度的处理方式。

- **标量**：使所有的数据点都设置相同的透明度。
- **线性数据**：使所有数据点设置透明度按照某个坐标轴方向线性变化。
- **矩阵**：使每个数据点选取不同的透明度。

上面的处理方式分别对应 MATLAB 中的 Alpha 函数中的参数。当参数是标量时，曲面的所有数据点都是相同的透明度。当 Alpha 函数中的参数是线性数据时，曲面的透明度按照某个维度的方向线性变化。

除了使用Alpha函数[illegible]设置曲面数据点的透明度之外，MATLAB还允许使用alim函数来设置透明度的上下限，将其上下限设置为$[A_{min},A_{max}]$。在MATLAB中设置透明度的上下限非常有用。[illegible]使用，MATLAB的[illegible]全称是Alpha，其参数只是对[illegible]设置Alpha属性的[illegible]。

6.6.18 透明控制实例

例6.53 在MATLAB中如[illegible]peaks函数的[illegible]。

Step 1 在MATLAB命令窗口中输入下面的命令

```
>> [x,y,z]=peaks(45);
>>surf(x,y,z);
>>shading interp
>>alpha(x);title('Along X')
>>alpha(y);title('Along Y')
```

Step 2 查看绘图结果。输入上面的[illegible]，按"Enter"键，得到[illegible]

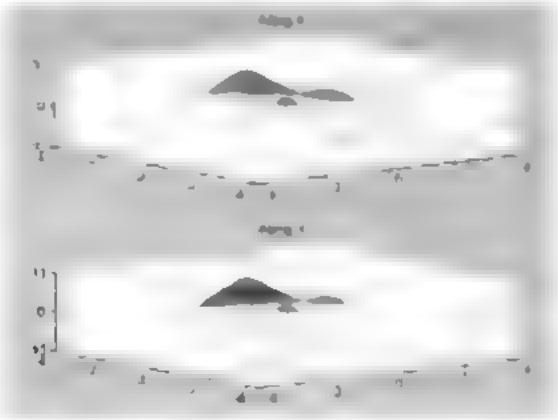


图6.63 设置不同的线条透明度

说明 从上面的[illegible]中可以看出，上面的[illegible]沿X轴不透明度[illegible]，下面的绘图的透明度则沿着Y轴不断加强。

例6.54 在MATLAB中绘制peaks函数的[illegible]，将绘图区域[illegible]透明，[illegible]部分设置为全透明。

Step 1 在MATLAB命令窗口中输入下面的命令

```
>> [x,y,z]=peaks(45);
>> surf(x,y,z);shading interp;
>> alpha(z)
>> Amin=-3;Amax=3;
>> alim([Amin,Amax])
>> alpha('scaled')
```

Step 2 查看绘图结果。输入上面的代码后，按"Enter"键，得到的图形如图6.64所示。

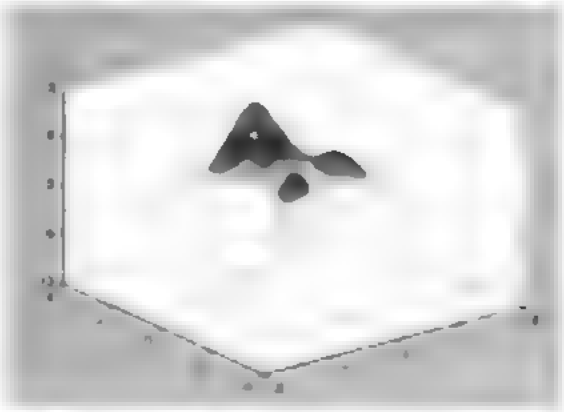


图6.64 设置图形上下两个部分的透明度

在MATLAB中，此函数alpha中的参数[illegible]，[illegible]结果[illegible]不[illegible]的透明度设置情况。alpha函数还可以通过其[illegible]，[illegible]的方法。参数方式是[illegible]'xdata'和'ydata'，[illegible]。在本例中，使用[illegible]上下两部分透明度[illegible]。[illegible]参考MATLAB的帮助文件。

例6.55 在MATLAB中显示Peaks函数的一维图形，[illegible]向上中部最透明，上下两部分则最不透明。

step 1 在MATLAB命令窗口中输入下面的命令

```
>> [x,y,z]=peaks(50);
>> surf(x,y,z);
>> shading interp
>> alpha(z)
>> alpha('interp')
>> alphamap('vdown')
```

step 2 查看图形结果，输入上面的命令后，按"Enter"键，[illegible]。

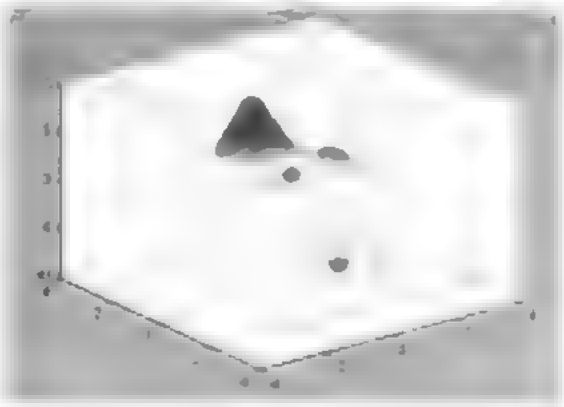


图6.65 设置三维图形的特殊透明度

在上面的例子中，使用alphamap命令来设置图形窗的透明度，参数值为vdown，[illegible]的透明度情况是由上向下逐渐透明。上下两部分则最不透明。关于alphamap命令的详细信息，可以参考MATLAB的相关帮助文件。

6.7 三维图形的简易命令

[illegible] ezsurfc等。

[illegible]

例6.56 [illegible] 等高线。

Step 1 [illegible]

```
>>ezmeshc('y/(1 + x^2 + y^2)',[-5,5,-2*pi,2*pi])
>>colormap jet
```

Step 2 [illegible]

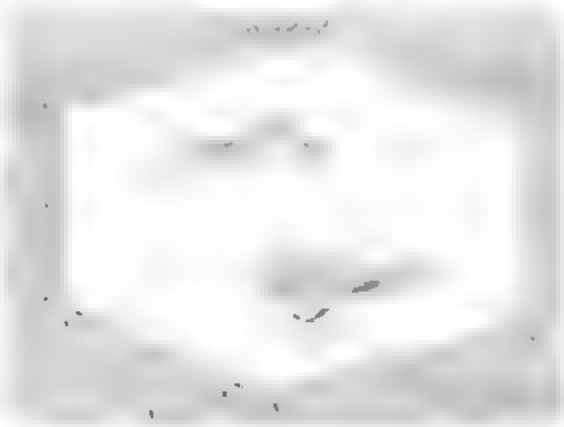


图6.66 绘制三维形形网格线以及等高线

[illegible]

例6.57 [illegible] 应的颜色和光照信息。

Step 1 在MATLAB命令窗口中输入下面的命令

```
>> shading flat
>> light('color','y','Position',[1 0 0],'Style','local')
>> colormap jet
>> view([-18,28])
```

Step 2 [illegible]

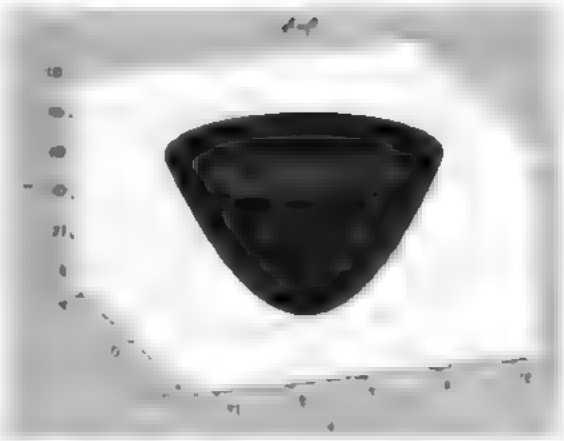


图 6.67 使用三维简易命令绘图

在使用简易命令绘图时，无论用户给定参数时，数字都是图形在“视域”上绘制，即数据点或是数量级，这样可以方便地绘制视域中的图形。

6.8 图形窗口

除了在 MATLAB 命令窗口中输入命令来创建图形外，还可以借用 MATLAB 提供的菜单栏。这是一个交互式的窗口，可以很方便地编辑各种图形。MATLAB 5.3 相对于之前的版本，在图形窗口的功能上有了较大的改进，用户可以在该图形窗口界面上实现几乎所有的编辑功能。

在命令窗口中输入绘图命令后，MATLAB 就会自动调用图形窗口，为用户的操作提供了解该窗口的创建方式及作用。因此，在本节的第一小节中将介绍如何创建和控制图形窗口，在后面的小节中将介绍如何利用图形窗口编辑 MATLAB 的图形。

6.8.1 创建和控制图形窗口

在 MATLAB 中，创建图形窗口的命令是 figure。该命令的常用调用格式如下：

- figure：创建一个图形窗口对象。
- figure('PropertyName','PropertyValue',...)：按照用户自定义的属性来创建一个图形窗口对象，用户可以对该图形窗口设置相应的属性。
- figure(h)：如果图形句柄 h 已经存在，则该命令会使得该图形窗口成为当前窗口；如果图形句柄 h 不存在，则创建一个句柄值为 h 的图形窗口对象。
- h = figure(...)：返回图形窗口对象的句柄。

如果希望了解当前或者一个图形句柄的窗口信息，可以使用下面的命令：

- get(h)：返回句柄值为 h 的图形窗口的参数名称以及当前数值。
- get(h,'name')：返回句柄值为 h 的图形窗口的参数名称，以便用户可以设置参数的当前数值。

例 6.58 在 MATLAB 中，使用相应的命令创建一个图形窗口对象。

Step 1 在 MATLAB 命令窗口中输入下面的命令：

```
>>figure(3)
```

step 2 查看图形效果。输入上面的代码后，按"Enter"键，得到的图形如图6.68所示。

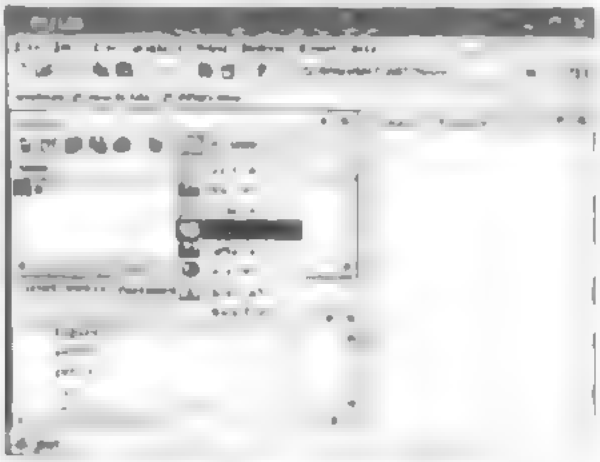


图6.68 创建图形窗口对象

说明 在默认情况下，如果用户只是输入命令"figure"，则会创建名为"Figure 1"的图形窗口。而在上面的程序代码中输入的命令为"figure(3)"，则创建的窗口名称为"Figure 3"。

例6.59 在MATLAB中，使用帮助的命令查看上面步骤中创建的图形窗口属性。

step 1 在MATLAB中的命令窗口中输入"get(3)"，得到下面的窗口信息。

```
BackingStore = on
CloseRequestFcn = closereq
Color = [0.8 0.8 0.8]
.....
PaperPosition = [0.634517 6.34517 20.3046 15.2284]
PaperPositionMode = manual
PaperSize = [20.984 29.6774]
.....
Type = figure
UIContextMenu = []
UserData = []
Visible = on
```

step 2 在MATLAB中的命令窗口中输入"set(3)"，得到下面的窗口信息。

```
Alphamap
BackingStore: [ {on} | off ]
CloseRequestFcn: string -or- function handle -or- cell array
.....
Tag
UIContextMenu
UserData
Visible: [ {on} | off ]
```

[illegible]

6.8.2 使用工具栏编辑图形

[illegible]

例6.60 [illegible]

Step 1 [illegible]6.69所示。

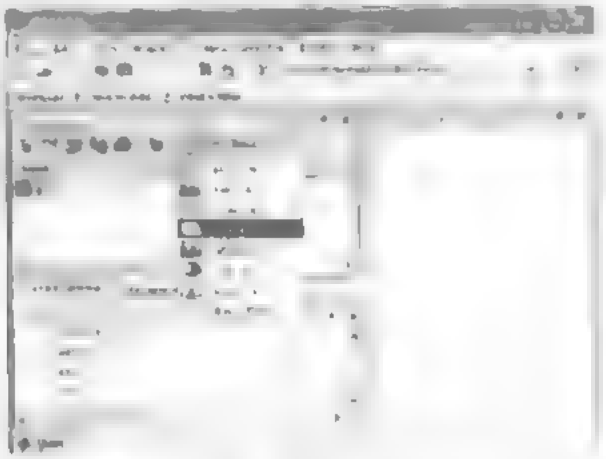


图6.69 利用窗口菜单创建图形

[illegible]的图形类型后，MATLAB会自动创建图形，如图6.70所示。

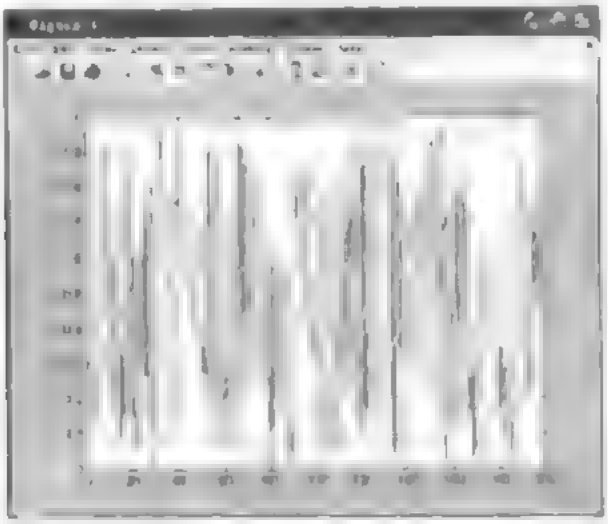


图6.70 创建图形窗口以及窗口图形

step 2 查看局部数据变化情况。[illegible]窗口工具栏中的“放大”按钮，然后选择需要放大的局部数据，在图形窗口中进行放大操作，如图6.71所示。

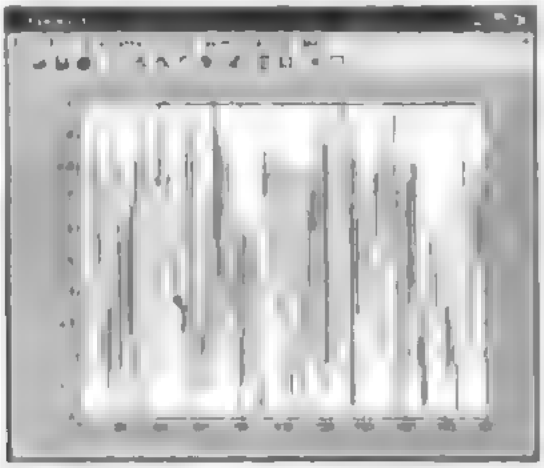


图6.71 放大局部数据

[illegible]，图形窗口[illegible]放大后的局部图形，在放大后的图形中编辑相关参数，如图6.72所示。

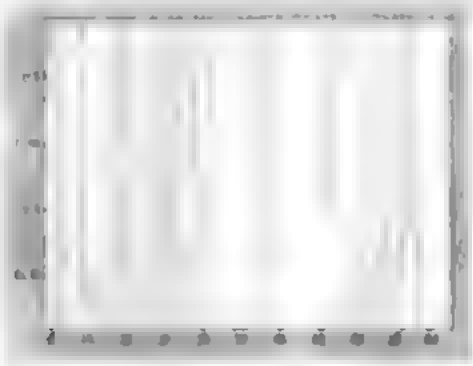


图6.72 放大后的局部数据

step 3 移动图形，查看[illegible]数据关系。由于在上面步骤中放大了局部数据，图形中[illegible]轴的[illegible]为[illegible]，如果需要查看其他数据节点的关系情况，就需要在图形窗口中移动图形，选择工具栏中的[illegible]按钮，移动图形，如图6.73所示。

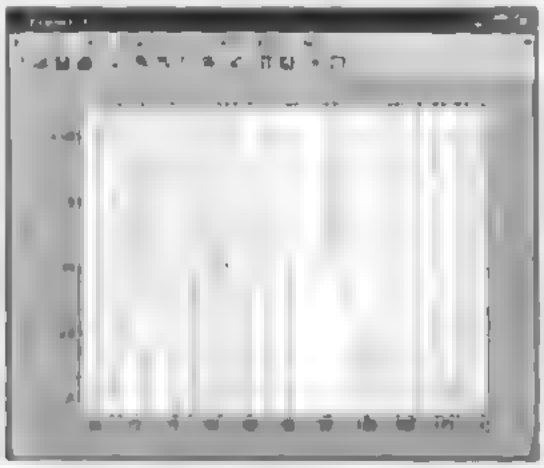


图6.73 移动图形

[illegible]

step 4 查看图形数据点的坐标值。选择工具栏中的数据光标按钮，然后在图形窗口中选择需要查看的数据点，查看该数据点的坐标值，如图6.74所示。

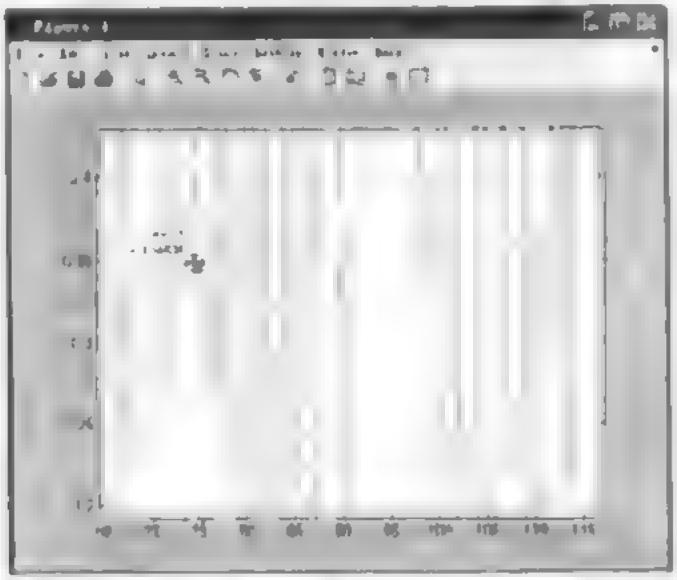


图6.74 查看图形数据点的坐标值

[illegible]

step 5 修改坐标值的显示方式。默认情况下，图形中显示的是默认方式，用户可以修改坐标值的显示方式，选中图形窗中的一个坐标值框，然后单击鼠标右键，在弹出的快捷菜单中选择"Display style" ⇨ "Window Inside Figure"选项，如图6.75所示。

图6.75 修改坐标值的显示方式

[illegible]

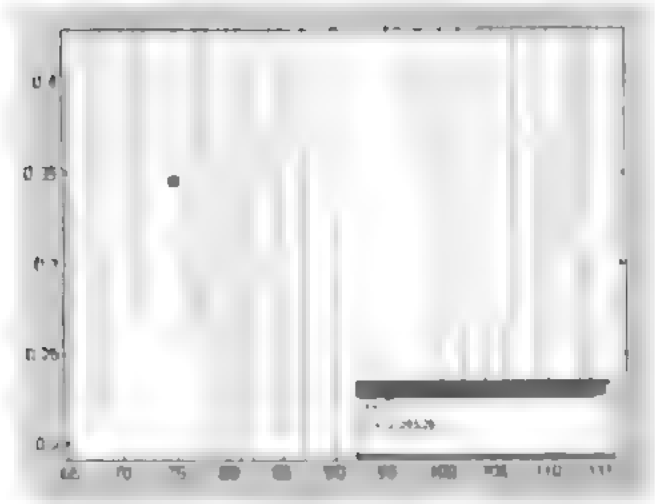


图6.76 修改后的显示方式

[illegible]

Step 6 [illegible]，如图6.77所示。

图6.77 添加图形文字注释

[illegible]

Step 7 [illegible]，如图6.78所示。

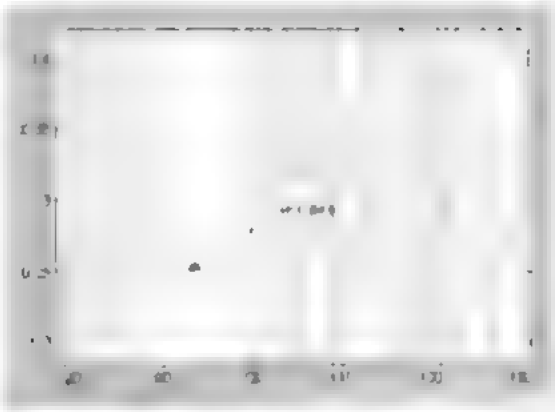


图6.78 移动图形窗口

[illegible]

step 8 [illegible]按钮[illegible]，再选择注释文件的固定点，如图6.79所示。

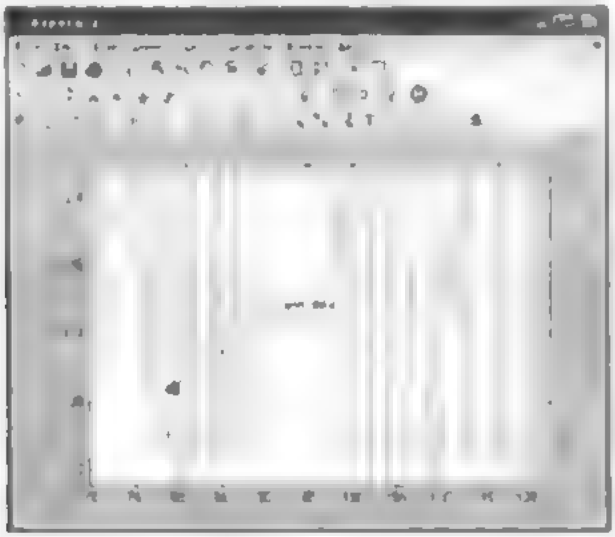


图6.79 固定图形的注释文件

[illegible]点上，如图6.80所示。

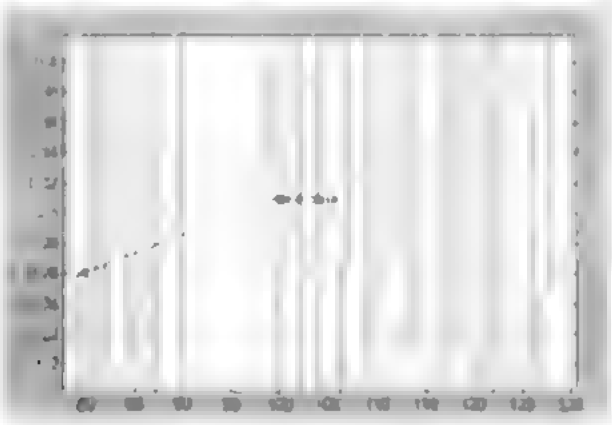


图6.80 移动图形窗口

6.8.3 使用绘图工具（plot tool）编辑图形

[illegible]

例6.61 [illegible]

step 1 [illegible]

```
>> z=peaks(45);
>> surf(z)
```

step 2 [illegible]

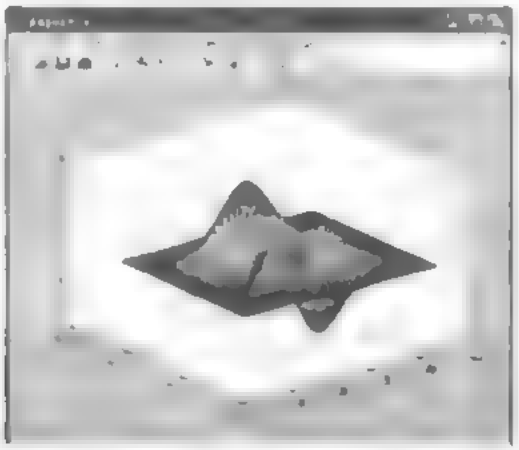


图6.81 创建的图形窗口

step 3 [illegible]

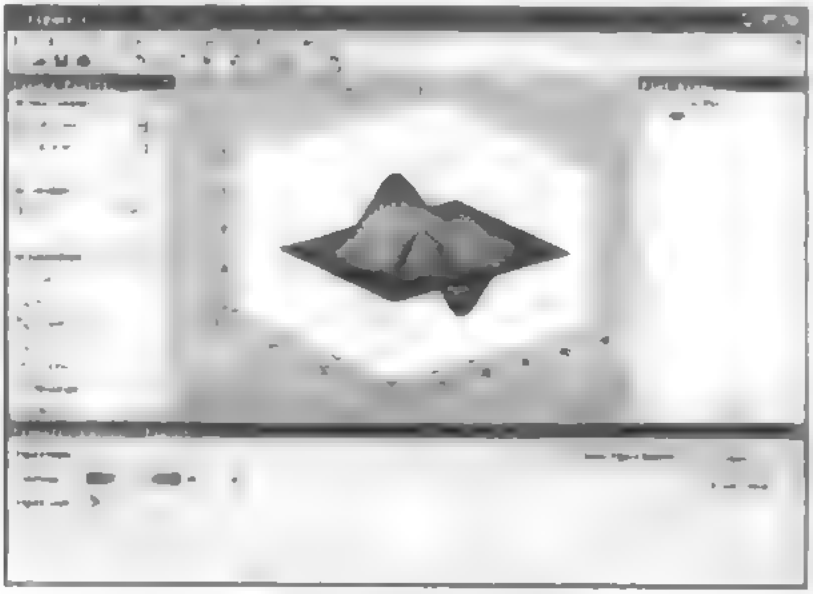


图6.82 打开绘图工具组件

从图示窗口中可以看出，给各个图形窗口的左、右侧和底部，其中是图窗口面板（Figure Palette）在图形窗口的左侧，绘图浏览器（Plot Browser）在图形窗口的右侧，属性编辑面板（Property Editor）在图形窗口的底部。下面简单介绍这些面板在编辑图形中所起的作用。

- **图形窗口面板（Figure Palette）**：在该面板中编辑图形时，可以在图形窗口中添加新子图（New Subplots），这个子图不受限制，可以多加，缩放和移动子图。在该编辑面板中，还可以为图形窗口添加各种工具栏（add toolbar）。最后，可以在该面板中查看图形的变量。
- **绘图浏览器（Plot Browser）**：在该面板中，可以查看图形中的各个对象，如坐标轴、变量表格等图形对象等。同时，可以单击该面板中的“Add Data”按钮，来导入新的数据。
- **属性编辑面板（Property Editor）**：该面板编辑图形中的每一个对象，通过选择对象后，在该的属性列表，用户可以在该面板中设置选中的图形对象的各种属性。这是非常有点的一个工具，可以避免繁杂的程序性的产生和编辑图形的时间。

Step 4 修改坐标轴的刻度间隔。在绘图浏览器（Plot browser）中选择“Axes”对象，属性编辑面板（Property Editor）中会显示出所选的坐标轴对象。选择“X Axis”选项卡，单击“Ticks”按钮，在对“Edit Axes Ticks”对话框，选择其中的“X Axis”选项卡，然后选择“Step by”单选钮，在其后输入新的间隔“5”，单击“Apply”按钮，就可以修改图形中的X坐标轴的刻度间隔，如图6.83所示。

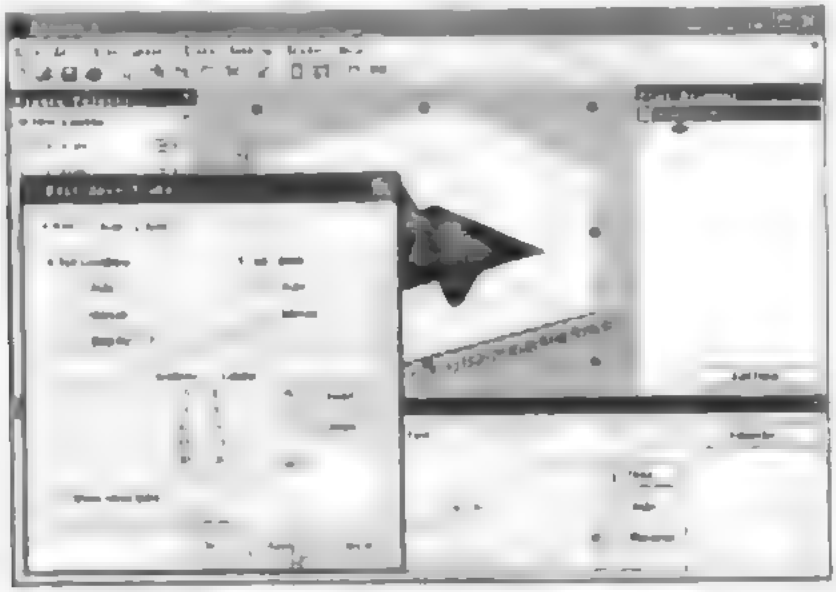


图6.83 设置坐标轴的刻度间隔

在此需要说明，关于图形中坐标轴的属性设置，一般需要使用图形的属性命令，但是使用图形窗口的绘图工具栏，可方便的完成对坐标轴的简单设置。

Step 5 修改其他坐标轴的刻度间隔，可以重复上面的步骤，设置坐标轴的刻度间隔，修改后的图形如图6.84所示。

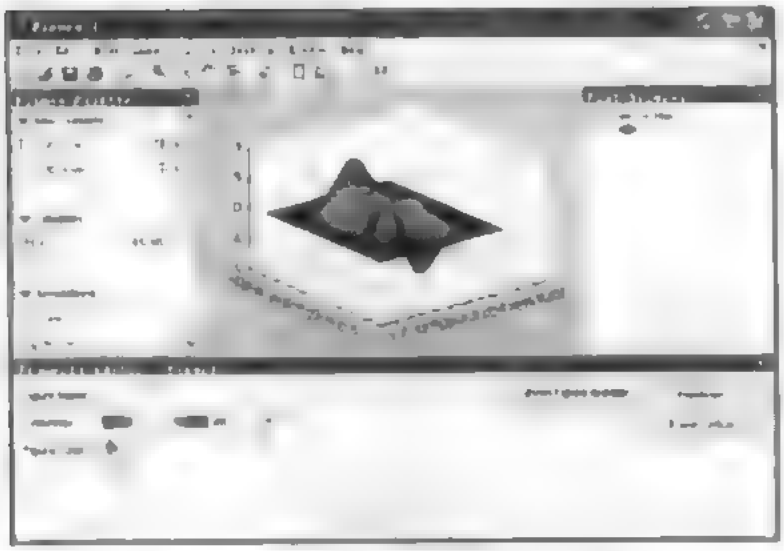


图6.84 修改坐标轴间隔后的图形

当修改坐标轴的刻度间隔后，X轴和Y坐标轴的刻度相比于原来图形中的刻度可能更密，时间会稍微变得更长。

Step 6 修改坐标轴的显示比例。在绘图浏览器（Plot Browser）中选择"Axes"对象，右键单击编辑面板（Property Editor），选择"X Axis"选项，选择"X Scale"选项卡的"Log"，同时选中"Reverse"选项，得到的图形如图6.85所示。

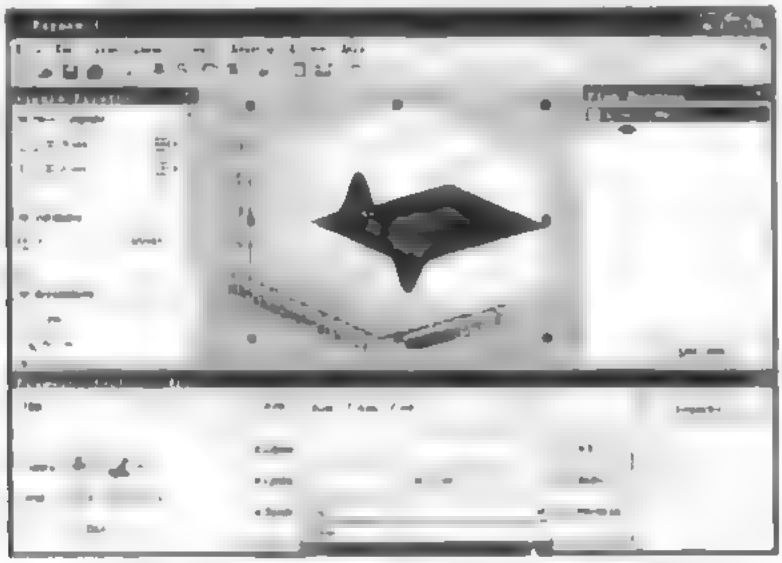


图6.85 修改坐标轴的显示比例

通过操作此时将会将X坐标轴设置为对数，并且反向显示。如果需要使用MATLAB的命令实现上面的结果，都需要使用相关的绘图命令。

拓展

除了上面这个图形的绘制外，MATLAB还提供了使用绘图工具编辑图形坐标轴的方法。对于图形的其他对象，都可以使用相关的方法来修改属性。

Step 1 [illegible]

```
>> y=cos(2*x)+sin(x);
```

Step 2 [illegible]

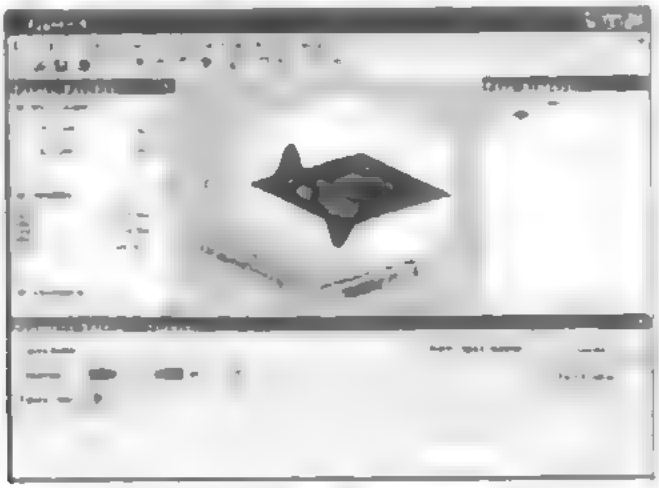


图 6.66　添加新变量后的图形窗口

[illegible]

Step 3 [illegible]

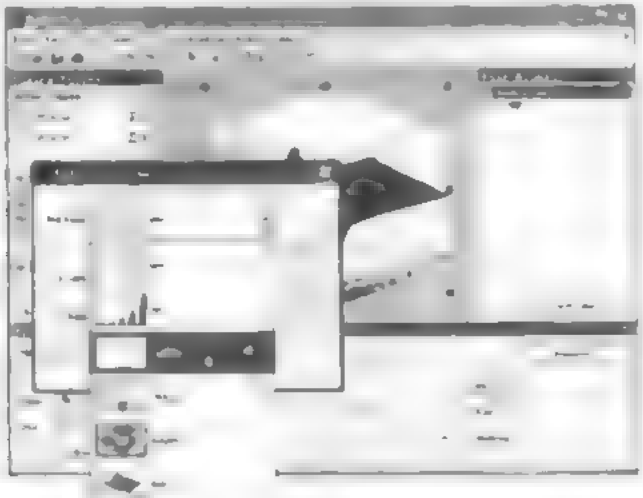


图 6.67　向图形中添加新的变量

[illegible]

Step 10 [illegible]

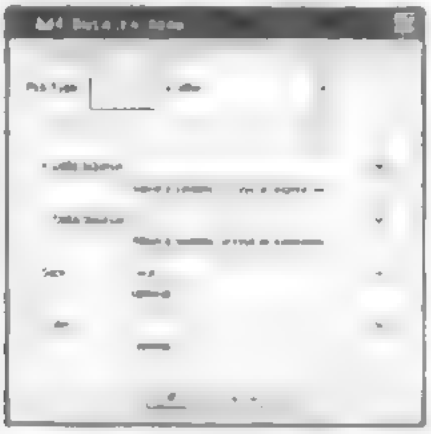


图 6.88 向图形中添加新的变量

[illegible]，MATLAB 就会在图形中添加该变量的散点图，如图 6.89 所示。

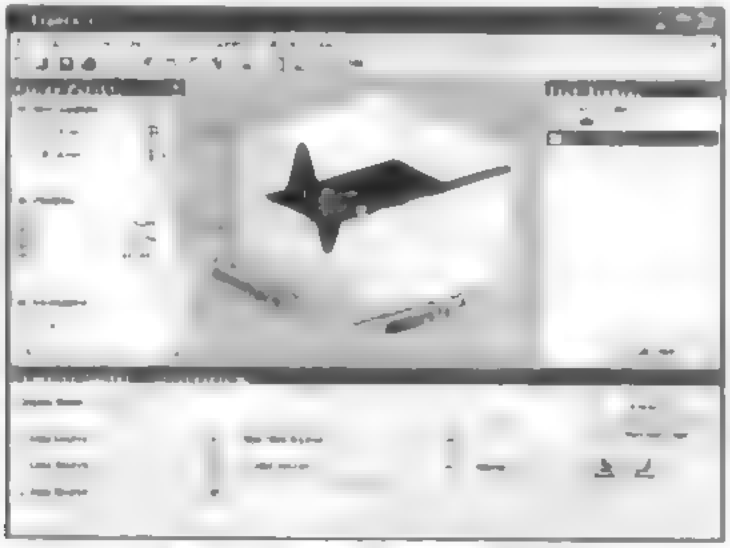


图 6.89 在图形中显示添加的变量

[illegible]

Step 11 [illegible]的添加子窗口，如图 6.90 所示。

图6.90 在图形中添加子图

在上面步骤的打开的窗格中，可以选择子类型格式。如果选择物理的二格，则会在图形中添加子图，在本步骤中，选择添加的是横向的子图，则会在图形窗口中显示如下图。

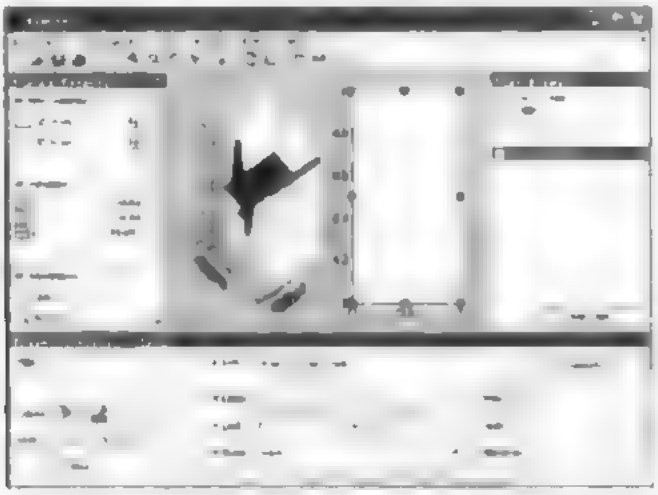


图6.91 在图形中添加新的坐标系

Step 12 在新子图的数据。使用步骤(7)中的方法为新坐标轴添加新的数据，在本步骤中选择的图表类型是"pie"，图表的数据来源为变量x和y，如图6.92所示。

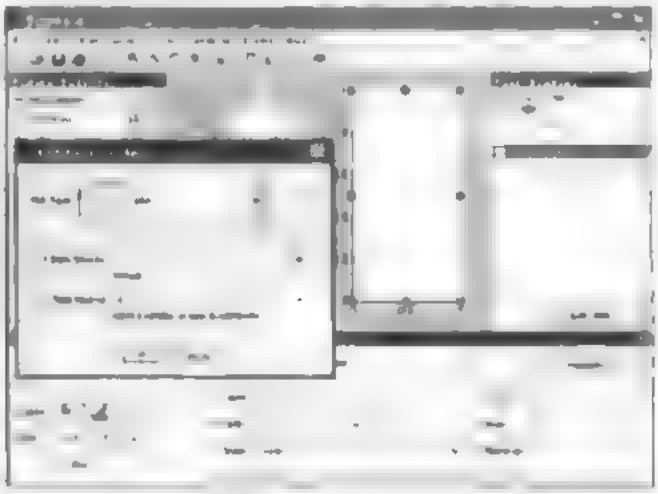


图6.92 添加子图的图形数据

[illegible]

step 13 [illegible]，单击"Add Graph to Layout"按钮中的"OK"按钮，查看上步骤中完成的子图，如图6.93所示。

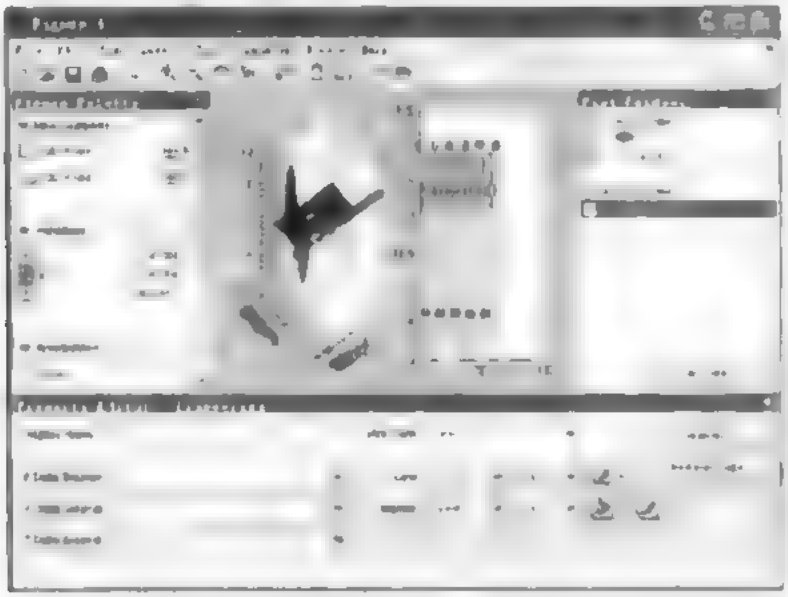


图6.93 上面步骤中添加的子图

[illegible]

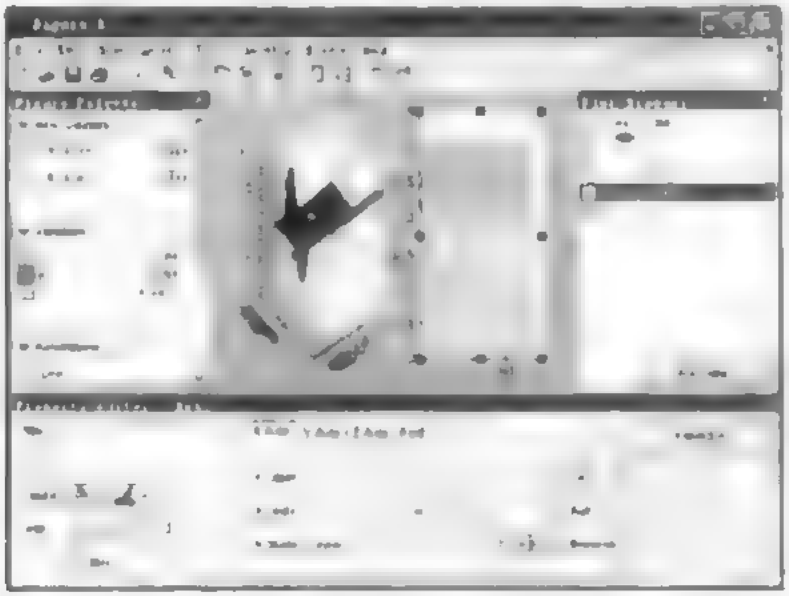


图6.94 修改坐标轴的刻度范围

step 14 [illegible]，[illegible]"Add Graph to Layout"选框中的"添加子窗口"，如图6.95所示。

图6.95 修改子图的显示方式

[illegible]

Step 15 [illegible]所示。

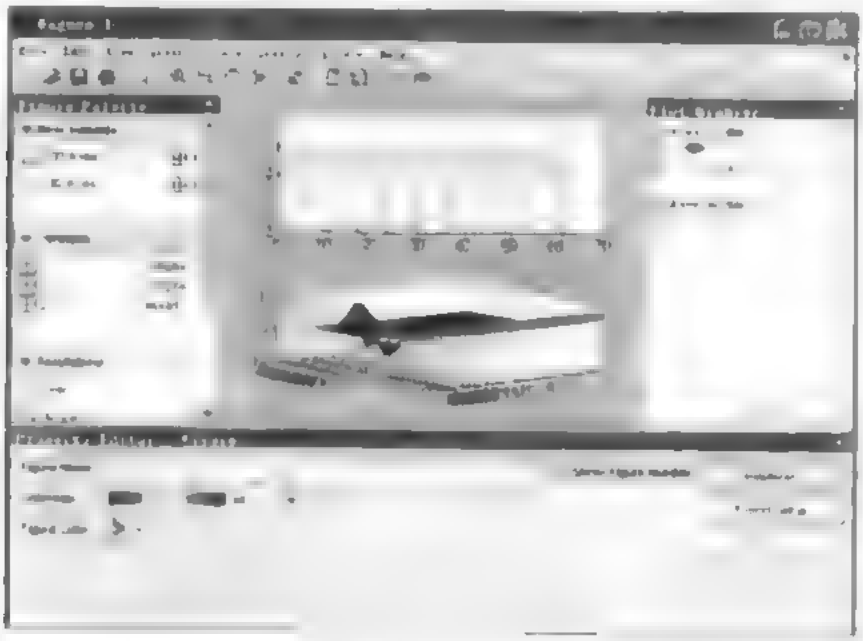


图6.96 修改后的子图

Step 16 [illegible]得到的结果如图6.97所示。

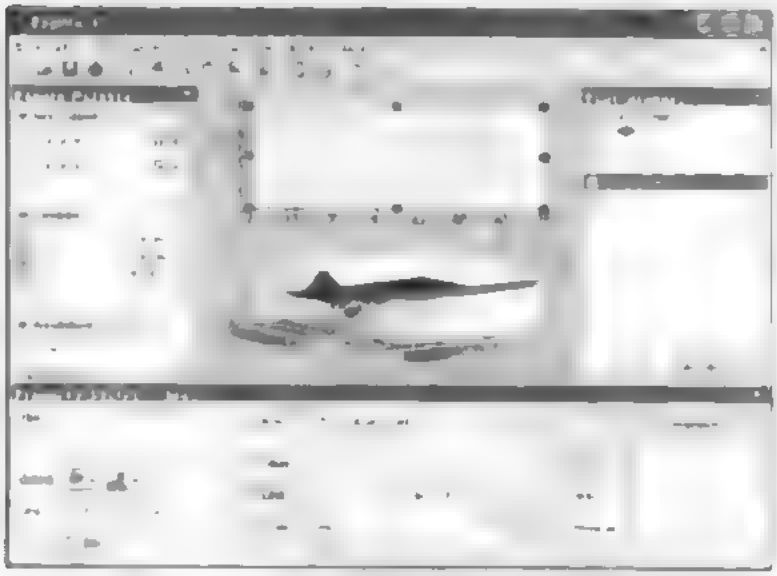


图6.97 修改子图的坐标轴范围

Step 17 [illegible]

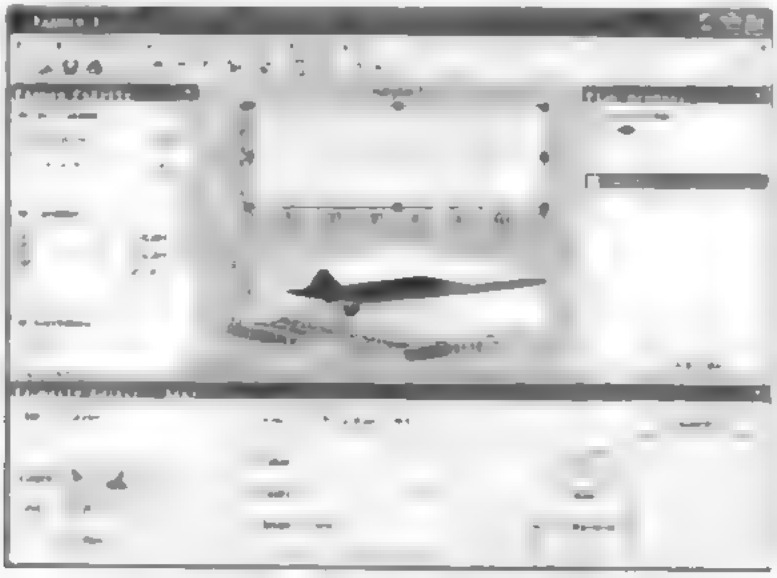


图6.98 设置子图的标题、边框

Step 18 [illegible]

[illegible]

图6.99 查看图形的其他属性

Step 19 隐藏绘图工具，绘图工具的主要作用是编辑图形，当图形处于编辑状态时，[illegible]的图形，要隐藏绘图工具，单击图形窗口的"Hide Plot Tools"按钮，隐藏绘图工具，如图6.100所示。

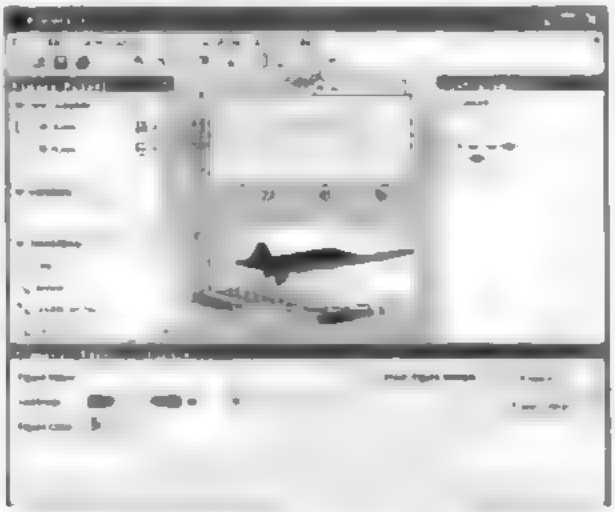


图6.100 隐藏绘图工具

当单击图形窗口的"Hide Plot Tools"按钮后，绘图工具窗口被关闭，[illegible]编辑窗口都会隐藏，这时编辑图形窗口大小，使其[illegible]，如图6.101所示。

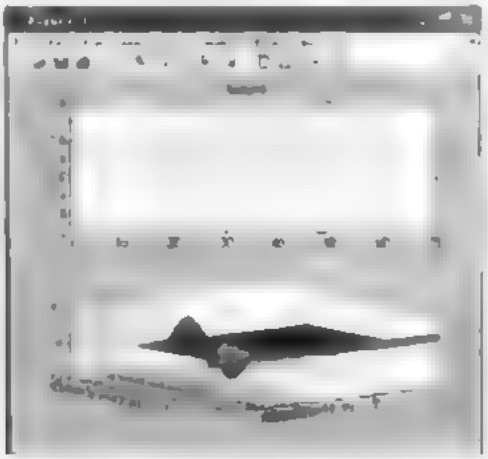


图6.101 隐藏绘图工具后的图形窗口

6.8.4 使用图形窗口进行数据分析

在[illegible]数据分析[illegible]节中，[illegible]数据分析的[illegible]，MATLAB 在绘图窗口增加进行数据分析的菜单选项。

例 6.62 使用 6.8.3 小节中的图形进行数据分析。

Step 1 [illegible] Statistics" 命令，如图 6.102 所示。

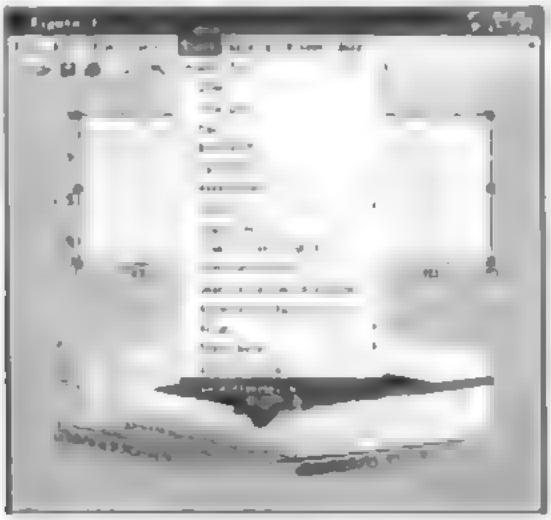


图 6.102 选择数据分析选项

Step 2 [illegible]，选择 x、y 的平均值，得到的图形如图 6.103 所示。

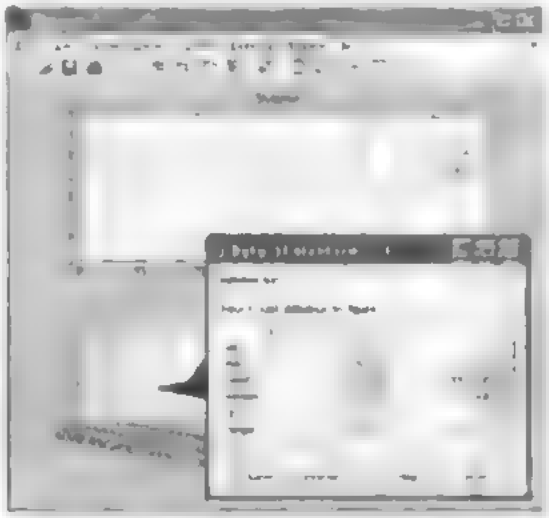


图 6.103 选择数据分析选项

[illegible]，MATLAB [illegible]

[illegible]

step 3 [illegible] fitting" 命令，如图 6.104 所示。

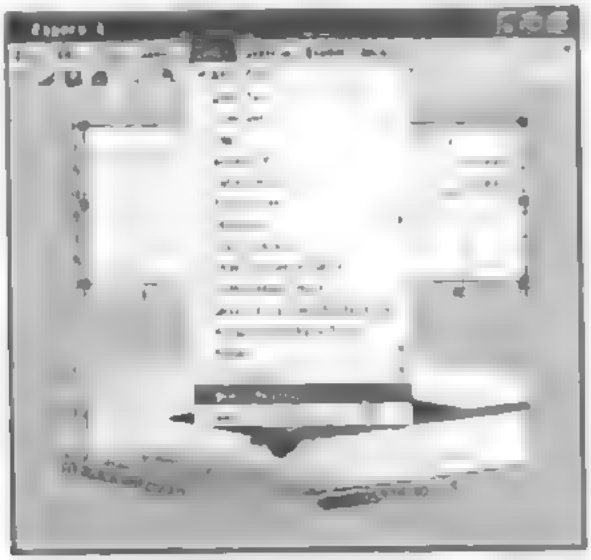


图 6.104　选择图形中的数据拟合

step 4 [illegible]

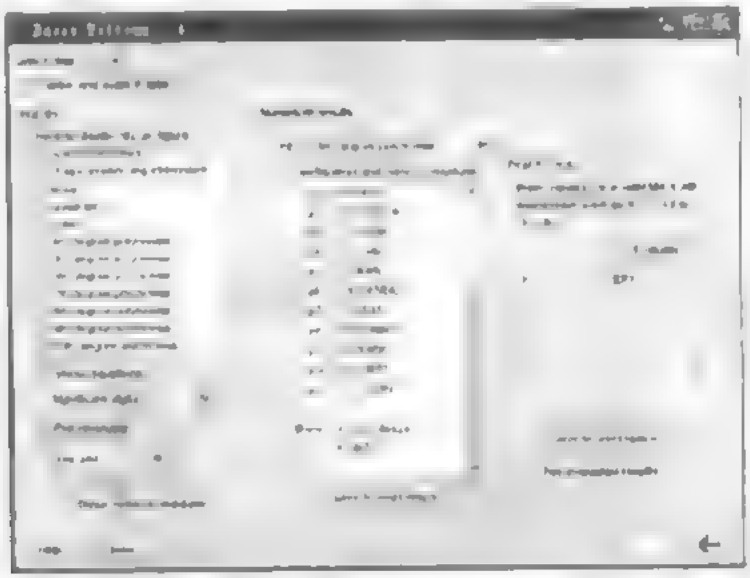


图 6.105　设置数据拟合的参数

[illegible]

356 ▶▶▶▶

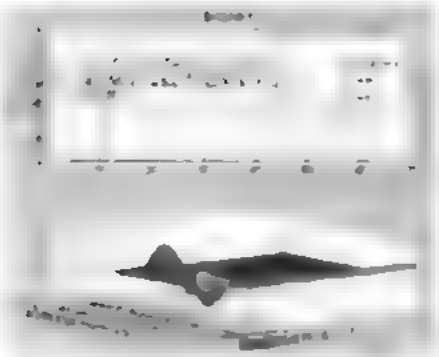


图 6.106 设置拟合参数后的曲线

Step 5 [illegible] 所示。

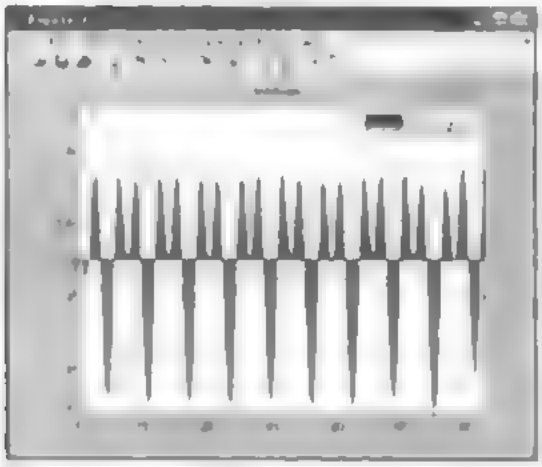


图 6.107 拟合的误差曲线

[illegible]

Step 6 [illegible] M-files...” 命令。如图 6.108 所示。

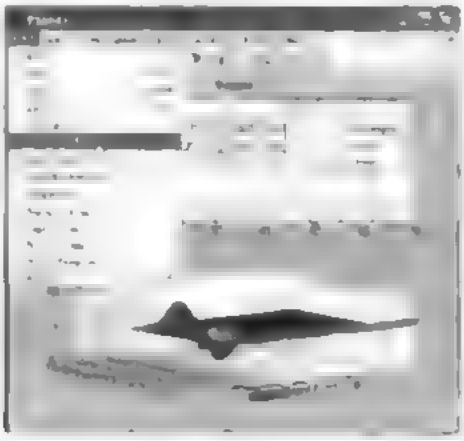


图 6.108 创建绘制图形的 M 文件

[illegible]，MATLAB会自动生成相应的M文件，如图6.109所示。

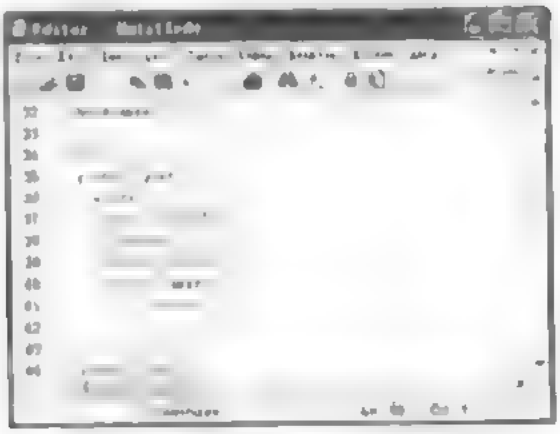


图6.109 对应的M文件

6.9 绘制复数变量图形

在MATLAB中，[illegible]复数变量[illegible]，MATLAB[illegible]。[illegible]，MATLAB[illegible]极坐复数[illegible]。

6.9.1 绘制复数图形原理

MATLAB[illegible]，[illegible]等，[illegible]的特点。

[illegible]，[illegible]，[illegible]。[illegible]，[illegible]，[illegible]，[illegible]，[illegible]颜色模式。

在MATLAB中，函数PLXGRID[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]。[illegible]

```
function z = cplxgrid(m)
t = (0:m)'/m;
theta = pi*(-m:m)/m;
z = t * exp(i*theta);
```

[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]法创建复数矩阵z，而且矩阵维度是$(m-1)\times(2m-1)$。

[illegible]

6.9.2 CPLXMAP 命令

CPLXMAP 命令可以用来绘制复数函数的图形，其语法是：CPLXMAP(z,w,B)。其中，参数 z 是正、负、[illegible]的复数。为了使用户能够绘制函数的图形，[illegible]

```
function cplxmap(z,w,B)
blue = 0.2;
x = real(z);
y = imag(z);
u = real(w);
v = imag(w);
if nargin > 2
   k = find((abs(w) > B) | isnan(abs(w)));
   if length(k) > 0
      u(k) = B*sign(u(k));
      v(k) = zeros(size(k));
      [illegible]
      [illegible]
   end
end
M = max(max(u));
m = min(min(u));
[illegible] M];
caxis([-1 1]);
g = ones(size(z));
[illegible]
hold on
[illegible]
hold off
[illegible]
```

在上面的程序代码中，[illegible]复数函数值的w[illegible]，然后确定[illegible]，使用mesh[illegible]，[illegible]，[illegible]。[illegible]

[illegible]

6.9.3 CPLXMAP 命令图形实例

例 6.63 使用CPLXMAP绘制复数函数的三维图形。

Step 1 在 MATLAB 命令窗口中输入[illegible]命令

```
>> z = cplxgrid(45);
>> cplxmap(z,z.^3)
>> title('z^3')
>> colorbar
```

step 2 在命令窗口中输入[illegible]，按"Enter"键，[illegible]

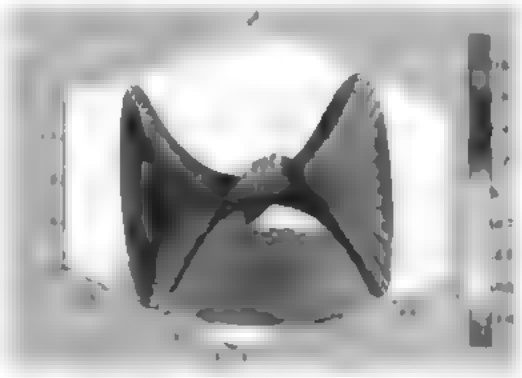


图 6.110 完成的复数图形

在上面的表示中，每个[illegible]复数函数 $f(z)=z^3$ 的[illegible]，并且[illegible]的虚部数值进行染色的。

6.9.4 CPLXROOT 命令

在 MATLAB 中，提供了 CPLXROOT 函数来绘制复数函数 $f=z^n$ 的 n 个根的空间曲面图，其[illegible]M 文件如下：

```
[illegible]
if nargin < 1, n = 3; end
if nargin < 2, m = 20; end
r = (0:m)'/m;
theta = pi*(-n:m:n*m)/m;
[illegible]
[illegible]
surf(real(z),imag(z),real(z),imag(z))
```

使用上面的程序[illegible]，[illegible]比较复杂的复数数值解的曲面图。下面使用一个简单的实例来说明。

6.9.5 CPLXROOT 命令图形实例

例 6.64 使用 CPLXROOT 函数绘制复数立方根的一组图形。

step 1 在 MATLAB 命令窗口中输入下面的命令：

```
[illegible]
>> view(-37.5,30)
>>cplxroot(3)
>>title('z的立方根')
```

step 2 按回车键[illegible]，按"Enter"键，[illegible]

[illegible]

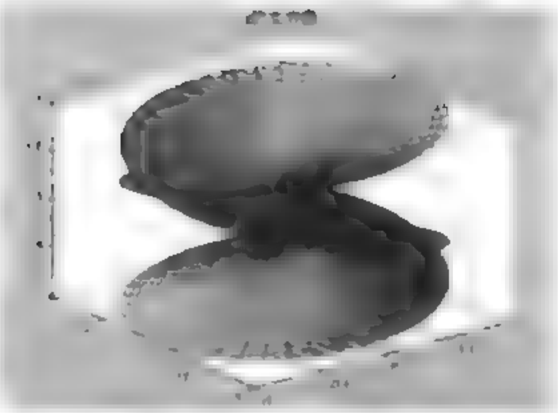


图 6.111 复数立方根的图形

6.10 图形的打印和输出

在完成了图形[illegible]输出，或[illegible]其他[illegible]，或者使用其他[illegible]软件[illegible]编辑。在 MATLAB 中，需要完成图形的打印和输出，可以使用图形窗口的菜单进行操作，也可以使用命令窗口中的命令。

使用菜单方式完成图形的打印和输出，比较简单、直接；使用命令方式完成图形的打印和输出，比较[illegible]，[illegible]在 M 文件中调用相应的命令，自动完成图形的打印和输出工作。

6.10.1 图形打印的菜单操作方式

[illegible]，需要[illegible]基本的操作方式，下面详细介绍相应的操作步骤。

例 6.65 演示如何打印例 6.48 所绘制的三维图形。

step 1 [illegible]，选择图形窗口中的 "File" | "Page Setup" 命令，打开 "Page Setup" 对话框，在其中对打印效果进行页面设置，如图 6.112 所示。

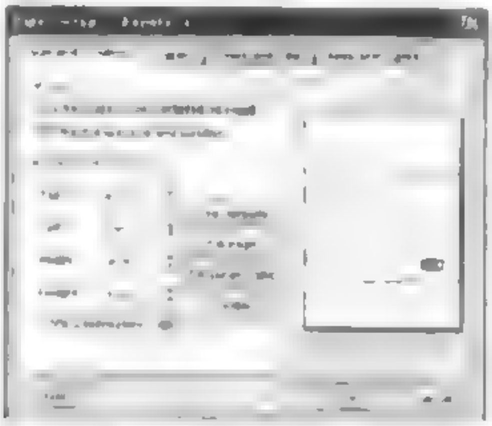


图 6.112 设置打印页面

[illegible]，在 MATLAB [illegible]，[illegible]，[illegible]，[illegible]

[illegible]

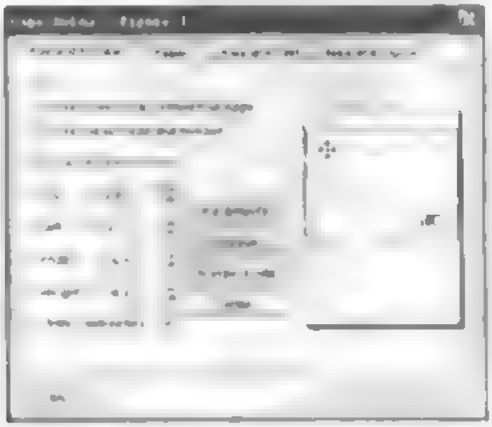


图 6.113　移动图形的位置

[illegible]

[illegible]

Step 2 [illegible]

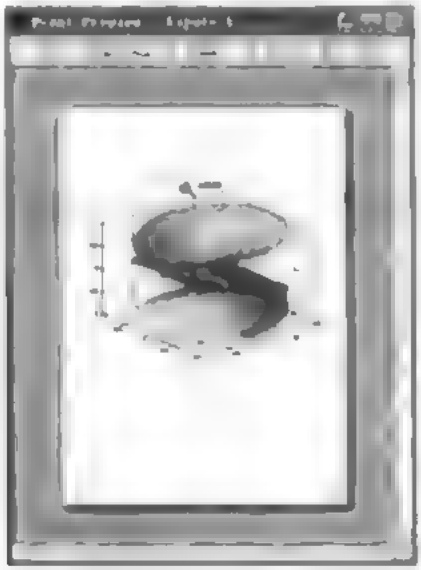


图 6.114　图形缩放

当鼠标指针指在图形的[illegible]，鼠标指针自动变成放大镜。当单击鼠标左键时，可以放大局部的图形。当单击鼠标右键时，可以缩小局部的图形。

Step 3 设置图形的标题。单击"Print Preview"对话框中的"Header..."按钮，打开"Figure Page

[illegible]

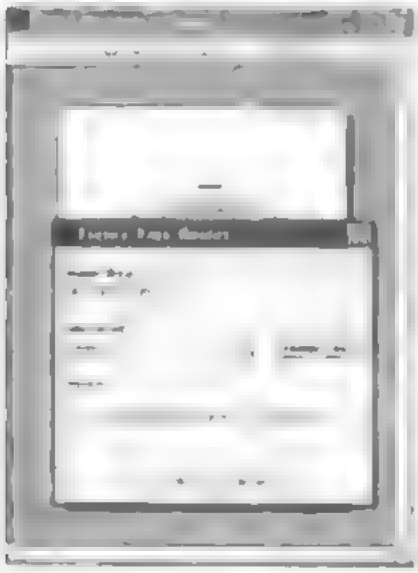


图 6.115 设置图形的标题

[illegible]

[illegible]

step 4 [illegible] 6.116 所示。

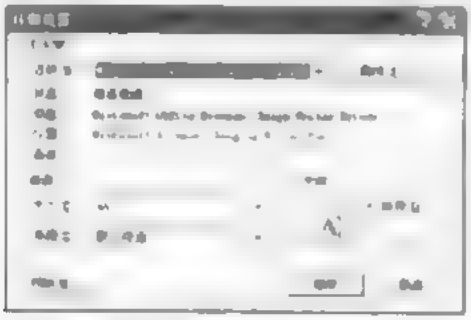


图 6.116 设置打印机的属性

[illegible] "Files" ⇒ "Print" 命令，直接打印该 MATLAB 的图形。

6.10.2 图形打印的命令操作方式

在 MATLAB 中，关于图形打印的常见调用命令如下。

- print 将图形发送到由 printopt 定义的打印设置和系统打印命令中。
- print filename　将图形输出到文件 filename 中。如果 filename 没有扩展名，print 命令自动选择一个扩展名。
- print -ddriver　使用由 ddriver 定义的打印设置打印当前图形。
- print -dformat　将当前图形复制到系统粘贴板上。
- print -dformat filename　以用户自定义的图形格式将图形输出到用户自定义的 filename 文件中。
- print -smodelname　打印当前 simulink 模型 smodelname。
- print ... -options　定义打印选项。
- [pcmd,dev] = printopt　返回当前系统的打印命令到字符串变量 pcmd 和输出设备到变量 dev 中。

下面简单使用一个实例，说明如何在 MATLAB 中合理运用上面的命令。

例 6.66　使用 MATLAB 的命令操作方式，打印 y=sinx 函数的图形。

在 MATLAB 命令窗口中输入下面的代码：

```
>> t=(1:100)/100*4*pi;
>> y=sin(t);
>> plot(t,y);
>> title('Print Figure')
>> print
```

在输入上面的代码后，图形会在 Windows 打印程序的管理下通过默认的打印机输出。

小结

本章从各个方面介绍了在 MATLAB 中如何实现数据和函数的可视化，主要介绍了如何绘制二维和三维图形，以及如何设置图形外观的各种属性：颜色、光照、送明、材质等。熟练掌握本章中的常见命令，可以根据需要绘制各种个性化的图形。在后面的章节中，将介绍如何在 MATLAB 中进行程序设计。

◆　◆　◆

第7章 MATLAB7.0基础编程

本章包括

- M文件编辑器
- MATLAB的程序结构
- 向量化
- 程序的调试和修改
- MATLAB的变量和关系式
- 控制语句
- 变量传递

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

7.1 简单实例

[illegible]

7.1.1 编写函数文件

例7.1 [illegible]

Step 1 [illegible]

[illegible]

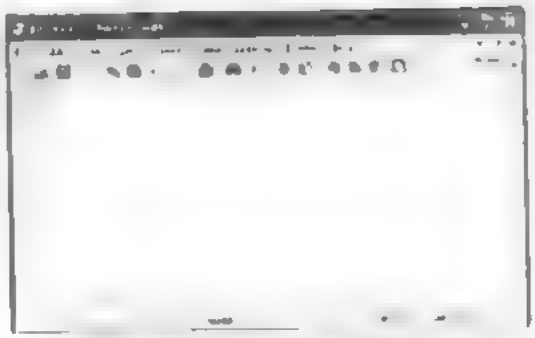


图 7.1　M 文件编辑器

Step 2 在 M 文件编辑器中，编[illegible]

图 7.2　编写 ssort 函数的脚本

上面函数程序的详细代码如下

```
function out=ssort(a)
% SSORT 程序代码按照升序排列数据
% [illegible]
% [illegible]
% Define variables:
% a                 Input array to sort
% ii                Index variable
% iptr              Pointer to min value
% nvals             Number of values in "a"
% out               Sorted output array
% temp              Temp varible for swaping
nvals=size(a,2);
for ii=1:(nvals-1)
    iptr=ii;
    for jj=ii+1:nvals
        if a(jj)<a(iptr)
            iptr=jj;
        end
    end
    if ii~=iptr
        temp=a(ii);
        a(ii)=a(iptr);
        a(iptr)=temp;
    end
```

end

Step 3 保存程序代码。[illegible]，或者选择编辑器工具栏中的[illegible]按钮，打开[illegible]文件，在其中保存上面步骤编写的程序代码，如图7.3所示。

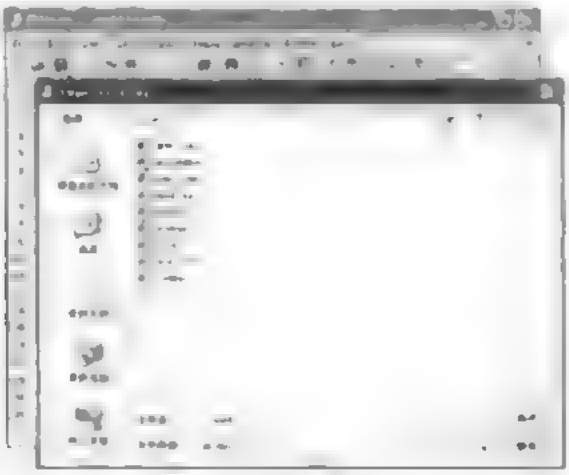


图7.3 保存用户编写的程序代码

[illegible]，MATLAB将[illegible]。在此[illegible]，[illegible]，[illegible]，[illegible]，[illegible]作为文件名，因此，本M文件的名称为ssort.m。

[illegible]，在M文件编辑器工具栏中选择[illegible]。在此[illegible]，[illegible]结果如图7.4所示。

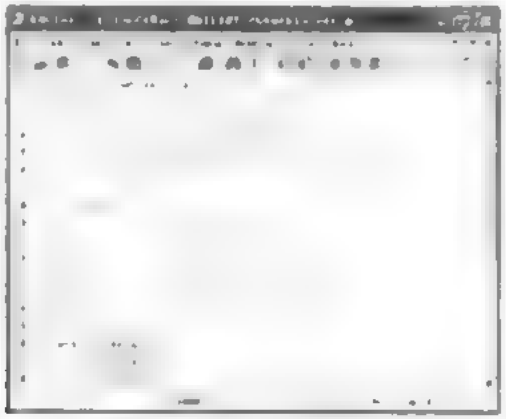


图7.4 保存后的程序代码

在[illegible]，[illegible]MATLAB[illegible]，[illegible]，[illegible]使用MATLAB[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]时，[illegible]。

7.1.2 编写脚本文件

在MATLAB环境中，M文件编辑器除了可以编辑函数文件外，还可以编辑脚本文件。在脚本文件中，除了没有定义函数名称之外，其他内容与前步骤类似，下面详细介绍。

例7.2 在MATLAB环境中，编辑对数据进行排序的脚本文件。

step 1 重新打开M文件编辑器，然后输入脚本文件的代码，如图7.5所示。

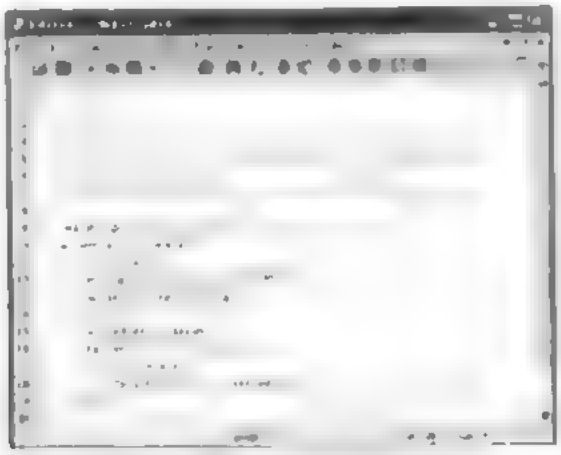


图7.5 输入脚本文件的代码

脚本文件的详细代码如下：

```
% Script file test_sort.m
%
% Purpose:
%   To read in an input data set,sort it into ascending order
%   using the "sort" function,and write the sorted data
%   to the command window.
%   The program uses function "sort" to do the actual sorting.
nvals=input('Enter number of the numbers to sort:');
array=zeros(1,nvals);
for ii=1:nvals
    string=['Enter value ' int2str(ii) ':'];
    array(ii)=input(string);
end
    sorted=sort(array);
    fprintf('\n sorted data:\n');
    for ii=1:nvals
        fprintf('%6.2f ',sorted(ii));
    end
```

step 2 将上面的脚本代码保存为"test_sort.m"文件，保存完毕后切换到工作目录。

7.1.3 运行代码

前面两个小节分别完成了函数代码和脚本代码，在本小节中，可以运行和检测代码。由于在test_sort.m的M文件中，调用了sort函数，所以可直接运行test_sort文件，这样就将两个代码都检测了。下面详细介绍如何检测。

图7.6 保存脚本文件代码

例7.3 运行并检测前面小节编写的数字排序程序。

step 1 将保存程序的目录设置为MATLAB的当前目录，然后在MATLAB的命令窗口中输入"test_sort"，按"Enter"键，得到的结果如图7.7所示。

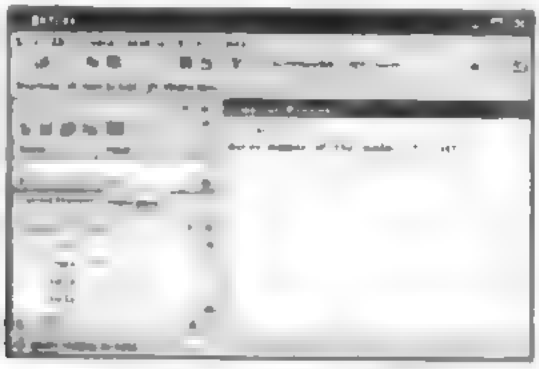


图7.7 检测程序代码

可以看出，在命令窗口中输入脚本文件名称"test_sort"，按"Enter"键后，MATLAB会自动调用脚本程序[illegible]。MATLAB的命令窗口中出现"Waiting for input"，表示系统处于接收数据的状态。

step 2 在程序的提示下，依次输入排序的数值，保存的结果如图[illegible]。

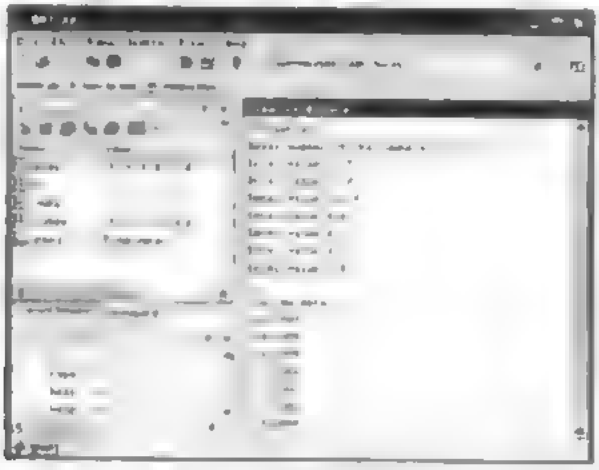


图7.8 运行程序代码

当输入相关的数值后，MATLAB会调用代码中的相应代码，[illegible]处理过程中的变量输入工作空间中。

step 3 重新输入变量数值，查看排序结果，[illegible]

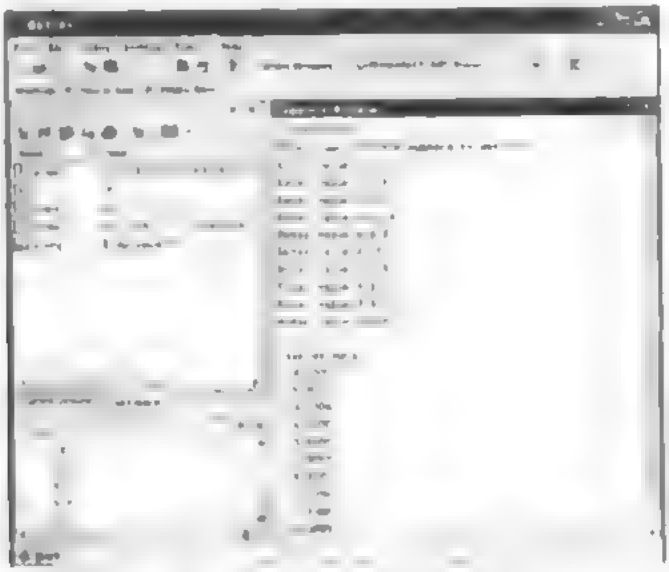


图7.9 重新运行程序代码

当在命令窗口中重新输入脚本文件的名称'test_sort'后，MATLAB再次调用[illegible]值，可以在代码中重新输入数值，得到新的排序结果。[illegible]，MATLAB[illegible]各个变量已经被覆盖成新的数值。

[illegible]，在编写代码过程中保留的[illegible]变量被称为脚本变量，存放在当前的工作空间中，[illegible]脚本结束后仍保留在变量表格中的变量，存放在MATLAB的工作空间中。

7.1.4 检测代码

接续上面小节的步骤。

step 1 查看两段程序代码的帮助信息。在命令窗口中依次输入'help ssort'和'help test_sort'，查看代码的在线帮助信息，得到的结果如下

```
>> help ssort
  SSORT Selection sort data in ascending order
  fuction ssort sorts a numeric data set into
  ascending order.Note that selection sort is
  relatavely inefficient. When sorting large data
  sets,please use MATALAB's "sort" function.

>> help test_sort
  Script file test_sort.m
    Purpose:
    To read in an input data set,sort it into ascending order
  using the Selection sort algorithm,and [illegible]
  to the command window.
```

```
This program calls function "ssort" to do the actual sorting.
```

[illegible]

[illegible]

STEP 2 查看M文件的详细代码，在命令窗口中输入命令"type ssort"或"type [illegible]"，查看对应M文件的详细代码，得到的结果如下

```
>> type ssort
function out=ssort(a)
% SSORT Selection sort data in ascending order
% fuction ssort sorts a numeric data set into
% ascending order.Note that selection sort is
% relatively inefficient. When sorting large data
% sets,please use MATALAB's "sort" function.

% Define variables:
% a                 Input array to sort
% ii                Index variable
% iptr              Pointer to min value
% [illegible]       [illegible] in "a"
% out               Sorted output array
% temp              Temp varible for swaping

[illegible];
for ii=1:nvals-1
    [illegible]
    for jj=ii+1:nvals
        [illegible]
            [illegible]
        end
    [illegible]
    if ii~=iptr
        [illegible]
        [illegible];
        [illegible];
    [illegible]
end
[illegible]
```

7.2 M文件编辑器

[illegible]M文件编辑器编写了基础的函数代码和脚本代码，本节将详细介绍该编辑器的使用方法和注意事项。

7.2.1 打开文件编辑器

M文件编辑器不会随着MATLAB的启动而启动，只有在用户编辑M文件时，该编辑器才启动。需要提醒读者的是，M文件编辑器不仅可以用来编辑M文件，还可以对M文件进行交互性调试。而且，M文件编辑器还可以用来读和编辑其他ASCII码文件。通常情况下，可以使用下面的方法来打开M文件编辑器。

- 在上面介绍过，可以单击命令窗口工具栏中的按钮，或者选择编辑栏中的"File"→"New"→"M-file"命令，打开M文件编辑器。这种方法适用于创建新的M文件。
- 如果需要编辑或者修改已经存在的M文件，可以单击MATLAB工具栏中的按钮，或者选择编辑栏中"File"→"Open"命令，打开Windows系统环境下的"Open"对话框，在对话框中选择需要编辑的M文件，然后单击对话框中的"打开"按钮，就可以打开该文件对应的编辑器。
- 除了在对话框中使用菜单选项打开M文件编辑器之外，还可以在命令窗口中输入"edit"命令，打开新的文件编辑器，或者输入"edit filename"打开一个存在M文件的编辑器，其中filename是M文件的名称，扩展名也可以不按扩展名。

[illegible]

例7.4 使用窗口命令打开前面创建的ssort文件。

将"…\MATLAB7.0\work"目录设置为MATLAB的当前目录，在命令窗口中输入">> edit ssort"命令，然后按"Enter"键，得到的图形如图7.10所示。

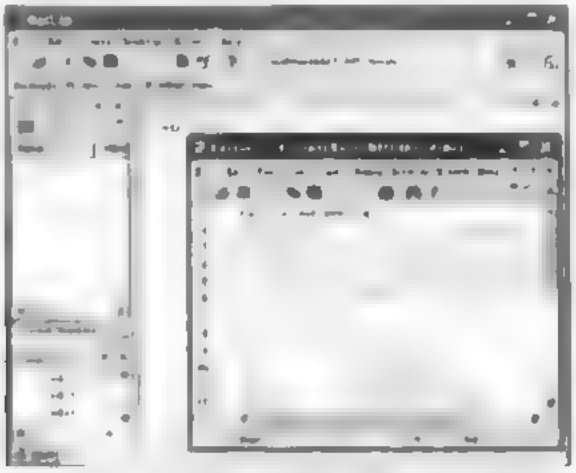


图7.10 打开ssort文件

7.2.2 设置M文件编辑器

有过编程经验的人员中间都有这样的经验，不同的编辑人员对编辑器的界面、字体、段落格式等都有自己的喜好，对此，MATLAB为用户提供了自定义这些属性的功能，用户可以选择相应的菜单选项来设置各种属性。

在本小节中，主要[illegible]设置M文件编辑器中的重要属性，[illegible]将[illegible]如何设置M文件的[illegible]、[illegible]、[illegible]设置M文件字体[illegible]，[illegible]在编辑M文件[illegible]，用户[illegible]需要[illegible]对应的程序代码文件。

下面以实例为单位，介绍如何进行编辑器的设置工作。

例 7.5 自定义设置M文件编辑器的属性。

step 1 在主[illegible]，单击[illegible]按钮，或者[illegible]编辑[illegible]“File”[illegible]“New”[illegible]“M-file”[illegible]，打开[illegible]M文件编辑器，然后选择[illegible]编辑器菜单中的[illegible]“File”[illegible]“Preferences”[illegible]，[illegible]“Preferences”对话框，[illegible]“Colors”选项，设置M文件编辑器中的各种字体颜色，如图7.11所示。

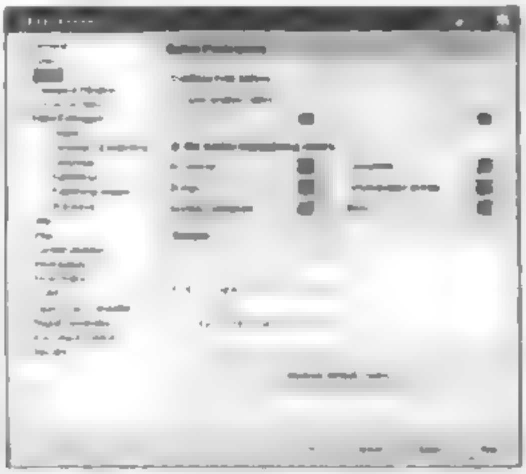


图7.11 设置M文件的各类字体颜色

在上面对话框的[illegible]，[illegible]程序语句中的关键字（keyword）、注释（comments）、字符串（strings）、[illegible]（unterminated strings）、系统命令（system commands）、[illegible]等，[illegible]一种字体颜色，[illegible]根据[illegible]来设置各种字体颜色，只需要在对应选项右侧的选项中选择对应的颜色。

step 2 设置M文件编辑器[illegible]，选择“Preferences”对话框中的“Editor/Debugger”选项下面的“Display”选项，选中“Enable datatips in edit mode”复选框，如图7.12所示。

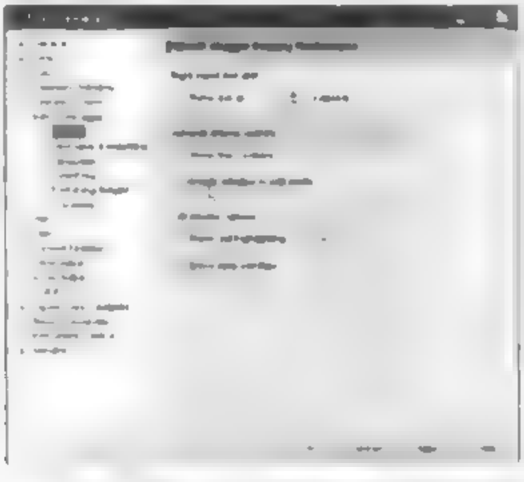


图7.12 设置M文件编辑器的显示选项

在默认情况下，MATLAB[illegible]“Enable datatips in edit mode”选项，[illegible]选择该选项

后，将光标移动到编辑器文件工具栏变量名称处，就会出现一个弹出菜单，显示该变量中存储的具体数据，这种设置有利于用户阅读程序代码。

step 3 设置M文件的保存选项。选择"Preferences"对话框中的"Autosave"选项，设置M文件的保存选项，如图7.13所示。

图7.13 设置M文件的保存选项

在上面的对话框中，可以设置M文件备份的方式。默认情况下，系统设置每隔一段时间保存正在编辑的程序代码，用户可以根据自己的情况来修改这个数值。

[illegible]，在加入[illegible]设置之后，单击"Apply"按钮，就可以将自定义的设置应用到当前编辑器文件的代码中。设置完成后单击选项窗口的"OK"按钮，就可以退出设置对话框。

7.2.3 设置M文件编辑器的打印属性

例7.6 自定义设置M文件编辑器的打印效果属性。

step 1 在上面[illegible]打开的M文件编辑器中，选择菜单"File"→"Page Setup"命令，打开"Page Setup"对话框，选中"Layout"选项卡，设置M文件的版面布局，如图7.14所示。

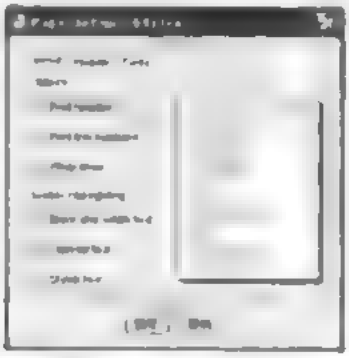


图7.14 设置M文件的版面布局

在默认情况下，MATLAB会选择"Print Preview"选项，[illegible]

Step 2 设置版头属性。选择"Page Setup"对话框中的"Header"选项卡，设置M文件的页眉版头属性。如图7.15所示。

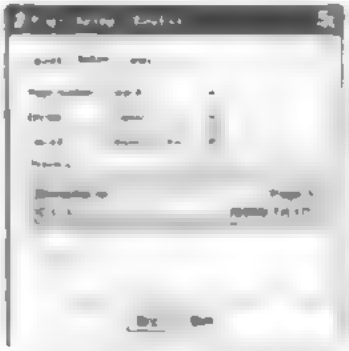


图7.15 设置版头属性

在[illegible]中，[illegible]

Step 3 设置M文件的字体属性。选择"Page Setup"对话框中的"Fonts"选项卡，设置M文件的打印字体属性，如图7.16所示。

图7.16 设置打印的字体属性

在默认情况下，MATLAB会选择"Use Editor Font"选项，这样打印出的文件的字体会采用M文件编辑器中的字体形式。如果希望使用自己设置的字体，可以首先选择"Use Custom Font"选项，然后在出现的下拉菜单中选择相应的字体信息。

7.3 MATLAB的变量和关系式

在MATLAB中，[illegible]

[illegible]

7.3.1 M文件的变量类型

在每个的程序结构中，变量是各种程序结构的基础。通常，MATLAB中的变量也有自己的命名规则，必须以字母开头，之后可以是任意字母、数字或者下划线，中间变量名不能有空格，变量名称区分大小写。最后，在MATLAB7.0中，变量名称不能超过63个字符，第63个字符之后的部分都将被忽略。

在MATLAB中有一些默认的预定义变量，用户在设置变量时应该尽量避免和这些默认的变量名一样，否则会造成系统和非预期的错误。表7.1列出了某几个预定义变量。

表7.1 MATLAB中的预定义变量

预定义变量	含义
ans	计算结果的默认名称
eps	计算机的零阈值
Inf(inf)	无穷大
pi	圆周率
NaN(nan)	表示非数或未定义的数值

在编写程序代码的时候，可以定义全局变量和局部变量两种类型，这两种变量类型在程序运行中有着不同的应用范围和工作原理。因此，有必要了解这两种变量的使用方法和特点。

当每一个函数在运行的时候，都会与有独立的工作空间，这个工作空间独立于MATLAB的基本工作空间和其他函数的工作空间。这样的工作原理保证了不同的工作空间中的变量相互独立，不会相互影响，这些变量都被称为局部变量。

在默认情况下，如果用户没有特别声明，函数运行过程中使用的变量都是局部变量。如果希望能够多个函数共享，可以使用全局变量。在MATLAB中，定义全局变量需要使用命令"global"，其调用格式如下：

```
global Var1 Var2
```

通过上述变量的命令，就可以使MATLAB开辟出一个一个函数空间以及基本工作空间共享的变量，每个希望共享全局变量的函数或者MATLAB基本工作空间中，都要对这个变量进行专门的定义。

如果某个函数在运行过程中修改了全局变量的数值，则其他函数空间以及基本工作空间的同名变量数值也会随之变化。

[illegible]

7.3.2 M文件的关键字

在命令变量名称时，MATLAB保留了一些关键字并且不允许用户以其进行命名，因此在定义变量名称的时候，应该避免使用这些关键字，否则系统会因为其相关操作数之类的错误提示。在MATLAB中，可以使用"iskeyword"命令来查看MATLAB中的关键字，例如结果如下：

```
>> iskeyword
ans =
    'break'
```

```
'case'
'catch'
'continue'
'else'
'elseif'
'end'
'for'
'function'
'global'
'if'
'otherwise'
'persistent'
'return'
'switch'
'try'
'while'
```

7.2 关系表达式

在MATLAB的常见分支或者循环控制结构中，经常会遇到判断结构，根据某种条件的数值0或者1而得出不同的结论，因此首先需要通过某种表达式来产生这种逻辑上的判断数值0或者1。在MATLAB中，能够产生这种逻辑数值0或者1的表达式有关系表达式和逻辑表达式。在本小节中，将详细介绍如何使用关系表达式。

关系表达式是针对两个变量的表达式，可能是两个数值变量或者字符串变量，通过表达式之间的关系得出逻辑值0（false）或者1（true），取决于两个变量之间的关系。

关系表达式的通用命令如下：

```
a op b
```

其中，a和b可以是算术表达式、变量、字符串等，op是一种逻辑关系。如果上面的表达式表达的关系是正确（true）的，则表达式返回数值1；如果表达式表达的关系是错误的，则返回数值0。

在表7.2中列出了MATLAB中常见的逻辑关系。

表7.2 MATLAB中的常见逻辑关系

关系运算符	含义
==	相等
~=	不等
>	大于
>=	大于等于
<	小于
<=	小于等于

例7.7 在MATLAB中，使用关系运算符进行运算，得到相应的结果。

step 1 在MATLAB的命令窗口中输入下列内容：

```
>> op1=(3<4);
>> op2=(3<=4);
>> op3=(3==4);
>> op4=(3>4);
>> op5=(4<=4);
>> op6=('A'<'B');
```

```
>> op=[op1;op2;op3;op4;op5;op6];
```

step 2 [illegible]，查看其数值，[illegible]结果如下

```
>> op
op =
     [illegible]
     [illegible]
     [illegible]
     [illegible]
     [illegible]
     [illegible]
```

step 3 查看逻辑运算数值的变量类型，得到的结果如下

```
>> class(op)
[illegible]
[illegible]
```

[illegible]

例7.8 在MATLAB中，以数组矩阵、数组、[illegible]进行关系表达式的运算。

step 1 在MATLAB的[illegible]中输入[illegible]

```
>> a=[-1,0;-2,1];
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
```

step 2 查看上面关系表达式的运算结果如下

```
>> Op1
Op1 =
     0     0
     0     1
>> Op2
Op2 =
     0     0
     0     1
```

step 3 将[illegible]进行关系运算，得到的结果如

```
>> Op3=(c>d)
??? Error using ==> gt
Matrix dimensions must agree.
```

[illegible]

[illegible]

在MATLAB中，[illegible]

7.3.4 关系表达式的优先级

例7.9 在MATLAB中关系表达式运算的优先级

step 1 在MATLAB的命令窗口中输入下列内容

```
>> Op1=4+5<2;
>> Op2=(4+5)<2;
>> S1=(Op1==Op2);
>> Op3=6-2>5;
>> Op4=(6-2)>5;
>> S2=(Op3==Op4);
```

step 2 查看[illegible]

```
>> A=[Op1,Op2,S1]
A =
     0     0     1
>> B=[Op3,Op4,S2]
B =
     0     0     1
```

[illegible]

尽管在MATLAB中，算术表达式的优先级高于关系表达式，在进行关系运算时不需加括号，就可以得出正确的结果。[illegible]

7.3.5 截断误差问题

例7.10 在MATLAB中对数值进行关系运算。

step 1 在 MATLAB 的命令窗口中输入下列内容：

```
>> a=0;
>> b=cos(3*pi/2);
>> op1=(a==b);
>> op2=(a~=b);
```

step 2 查看上面关系运算的结果如下：

```
>> op1
op1 =
     0
>> op2
op2 =
     1
```

根据上面程序代码的结果可以看出，尽管$b=\cos(\frac{3}{2}\pi)=0$，数值a也等于0，但是MATLAB却认为两个变量数值不相等，因此op1的逻辑数值为0，op2的逻辑数值为1。造成这样结果的原因是，MATLAB在数值运算中的截断误差（roundoff error）。在MATLAB中，表达式$\cos(\frac{3}{2}\pi)$得到的数值并不直接等于0，而是等于1.2246×10^{-16}，所以，当将变量a和b进行比较时，返回就是逻辑值0（false）。

step 3 修改程序代码，重新比较两个数值大小：

```
>> op3=abs(a-b)<1.0e-15
op3 =
     1
```

在上面的程序代码中，为了避免MATLAB中的数值运算截断误差，将两个数值a和b进行比较时，使用的表达式为$|a-b|<1.0\times10^{-15}$，也就是说，判断两个数值变量之间的数值间隔是否足够小，这个数值就可以避免MATLAB中的运算截断误差。

逻辑表达式

在MATLAB中，逻辑表达式是通过逻辑关系符将两个或者多个进行连接得到的逻辑值。在MATLAB中，提供三种比较常见的逻辑运算符：AND、OR和XOR。这三种逻辑运算符是二元关系运算符，同时，MATLAB还提供了一个非二元关系运算符NOT。

在表7.3中列出了MATLAB中常见的逻辑运算符。

表7.3 MATLAB中的常见逻辑运算符

关系运算符	含义
&	逻辑AND
|	逻辑OR
Xor	逻辑Xor
~	逻辑NOT

例7.11 在MATLAB中，使用逻辑表达式进行数值逻辑运算。

step 2 在MATLAB的命令窗口中输入下列内容：

```
>> value1=1;
>> value2=0;
>> value3=-10;
>> op1=~value1;
>> op2=value1|value2;
>> op3=value1&value2;
>> op4=value1&value2|value3;
>> op5=value1&(value2|value3);
>> op6=~(value1&value3);
>> op=[op1,op2,op3,op4,op5,op6];
```

step 2 查看逻辑运算的结果如下：

```
>> op

op =
     0     1     0     1     1     0
```

例7.12　在MATLAB中，综合使用逻辑和关系运算得出逻辑结果。

step 1 在MATLAB的命令窗口中输入下列内容：

```
>> a=2;
>>b=[-1,-2;0,10];
>>op1=~(a>b);
>>op2=(~a)>b;
>>op3=~a>b;
>> op=(op2==op3);
```

step 2 查看上面的运算结果如下：

```
>> op1
op1 =
     0     0

>> op2
op2 =
     1     1
     0     0
>> op3
op3 =
     1     1
     0     0
>> op
op =
     1     1
     1     1
```

上面的实例并不复杂，主要是为了介绍关系表达式和逻辑表达式的优先级问题。由于表达式中op2和op3是完全相同的，因此可以看出逻辑运算符~的优先级要高于关系运算符>。下面简要介绍MATLAB中关系运算符和逻辑运算符的优先级情况，包括算术运算符在内，其优先级依次为：

- 所有的算术运算符的优先级最高，高于关系和逻辑运算符；
- 其次是所有的关系运算符（==，~=,>,>=,<,<=），关系运算符从左向右运算；

- 接着是逻辑上的NOT运算符~；
- 接着是逻辑上的AND运算符&，运算顺序是从左向右；
- 最后是逻辑上的XOR运算符|，运算顺序是从左向右。

逻辑运算函数

在本小节中，将向读者介绍MATLAB中的逻辑运算函数，或者称为is类函数。当函数判断的条件成立时，该类函数将返回逻辑数值1；当函数判断的条件不成立时，该类函数将返回逻辑数值0。这些函数都可以用在关系表达式或者逻辑表达式中，也可以用来作为程序结构的判断条件。

表7.4列出了MATLAB中常见的逻辑函数。

表7.4 MATLAB中的常见逻辑函数

函数名称	功能
ischar	判断变量是否是字符串变量
isempty	判断变量是否是空白数组
isinf	判断数值变量是否是无穷大
isnumeric	判断变量是否是数值数组

MATLAB的程序结构

和其他编程语言类似，MATLAB也给用户提供判断结构来控制程序流的执行次序。一般来讲，决定程序结构的语句有顺序结构、分支结构和循环结构三种，每种语句结构都有各自的流控制机制，相互配合使用可以实现功能强大的程序。

由于MATLAB的这种控制命令用法和其他语言用法十分类似，因此本节只结合MATLAB的特点对控制命令进行简要的介绍。

顺序结构

顺序结构是最遵循逻辑思路的程序代码结构，批处理文件就是典型的顺序语句的文件，这种语句不需要任何特殊的流控制。这种顺序结构是最基础的程序结构，也是其他控制流语句中的重要组成部分。

例7.13 在MATLAB中，使用顺序结构编写绘制函数的图形。

step 1 单击MATLAB命令窗口工具栏中的□按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
%定义符号变量t和tao
syms t tao
%定义积分表达式
y=exp(-t/3)*cos(1/2*t);
%对表达式进行积分
s=subs(int(y,0,tao),tao,t);
%绘制积分图形
ezplot(s,[0,4*pi]);grid
```

step 1 单击M文件编辑器中的“保存”按钮，将上面的程序代码保存为“ezexm”，得到的结果如

到7.17所示。

图7.17 保存M文件程序代码

step 3 返回MATLAB的命令窗口，输入"[illegible]"，然后按"Enter"键，保存文件结束语句，如图7.18所示。

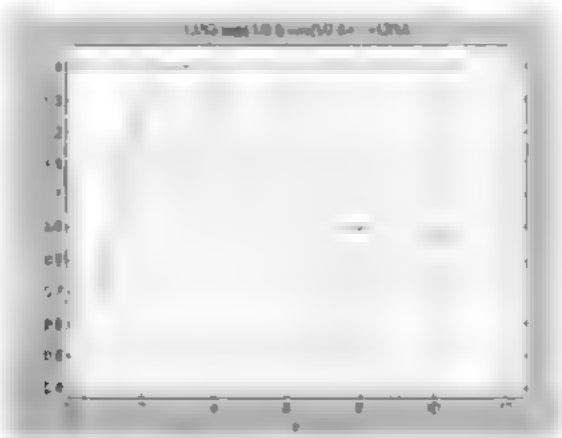


图7.18 得到的程序结果

在上面的程序代码中，首先定义符号变量，然后定义被积表达式，进行积分运算，最后调用ezplot命令绘制被积函数的图形。这样编写的程序代码不仅能够实现，而且容易阅读，容易理解，这是程序结构的重要优点。

在上面的vs.m文件程序中，使用了ezplot的绘图命令，[illegible]

7.4.2 if分支结构

当需要根据一定条件来执行不同的操作时，可以使用条件语句，在MATLAB中称作分支结构，或者称为是if-else-end语句。

根据不同条件情况，if分支结构有多种形式，其中最简单的形式是：当满足条件表达式时，执行一组MATLAB语句，包括一组命令statements，称为一个组命令，其格式为：

```
if expression
    statements
end
```

如果上面的expression是一个多数组，MATLAB以个条件表达式处理，在MATLAB中，if语句

结构的更为普遍的调用格式如下：

```
if expression1
    statements1
elseif expression2
    statements2
.....................
else
    statementsk
end
```

如果条件语句只有两种选择的可能，则其调用格式如下：

```
if expression1
    statements1
else expression2
    statements2
end
```

在大多数情况下，条件表达式会由关系表达式或者逻辑表达式组成，这些表达式返回的都是逻辑值0或者1，将作为条件判断的依据。为了提高程序代码执行的效率，MATLAB会尽可能少地检测这些表达式的数值。

例7.14 在MATLAB中，使用if分支结构编写求解一元二次方程$ax^2+bx+c=0$的程序代码，并且运行检测该代码结果。

step 1 分析分支结构的判断条件。根据基础数学知识，一元二次方程$ax^2+bx+c=0$的根的性质直接取决于判别式$\Delta=b^2-4ac$的数值。当$\Delta=0$时，该方程有两个相等的实根；当$\Delta>0$时，该方程有两个互不相等的实根；当$\Delta<0$时，该方程有两个虚根。

step 2 单击MATLAB命令窗口工具栏中的D按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
% script file calc_root.m
%
% purpose:
% This program solves for the roots of a quadratic equation
% of the form a*x^2+b*x+c=0.It calcaulates the answers of
% roots the equation posseses.
%
% Define variables:
% a       coefficient of x^2
% b       coefficient of x
% c       constant term
% x1      first root of the equation
% x2      second root of the equation

disp('This program solves for the roots of a quadratic equation');
disp('of the form a*x^2+b*x+c=0');
a=input('Enter the coefficient A:');
b=input('Enter the coefficient B:');
c=input('Enter the coefficient C:');
discriminant=b^2-4*a*c;

%如果判别式大于0
```

```
% 则根据二元方程的公式求出两个不同的实数解
if discriminant>0
    x1=(-b+sqrt(discriminant))/(2*a);
x2=(-b-sqrt(discriminant))/(2*a);
% 在命令窗口显示求解结果
    disp('This equation has two real roots');
    fprintf('x1=%f\n',x1);
fprintf('x2=%f\n',x2);

% 当判别式等于0，则返回两个相同的实数根
elseif discriminant==0
    x1=-b/(2*a);
    disp('This equation has two identical roots');
    fprintf('x1=x2=%f\n',x1);

% 当判别式小于0，则返回两个虚根
else
    real_part=-b/(2*a);
    image_part=sqrt(abs(discriminant))/(2*a);
    disp('This equation has two complex roots');
    fprintf('x1=%f+%i%f\n',real_part,image_part);
    fprintf('x2=%f-%i%f\n',real_part,image_part);
end
```

step 3 单击M文件编辑器中的“保存”按钮，将上述的程序代码保存为“calc_root.m”，保存结果如图7.19所示。

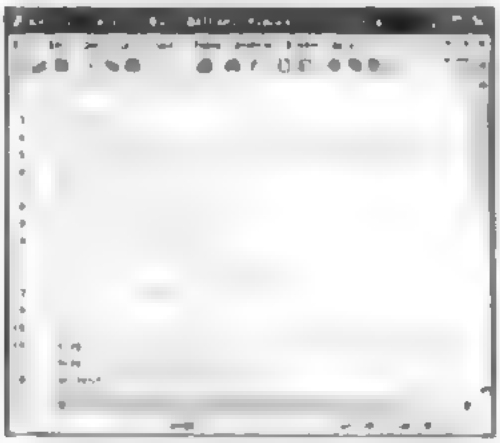


图7.19 保存程序代码

step 4 返回到MATLAB的命令窗口，输入“calc_root”，然后按“Enter”键，根据程序代码的提示，依次输入方程的系数，得到的结果如下

```
>> calc_root
This program solves for the roots of a quadratic equation
of the form a*x^2+b*x+c=0
Enter the coefficient A:1
Enter the coefficient B:5
Enter the coefficient C:6
This equation has two real roots
x1=-2.000000
x2=-3.000000
```

```
>> calc_root
This program solves for the roots of a quadratic equation
[illegible]
Enter the coefficient A:1
Enter the coefficient B:4
Enter the coefficient C:4
This equation has two identical roots
x1=[illegible]
>> calc_root
This program solves for the roots of a quadratic equation
of the form a*x^2+b*x+c=0
Enter the coefficient A:1
Enter the coefficient B:2
Enter the coefficient C:5
This equation has two complex roots
x1=-1.000000+i2.000000
[illegible]
```

在上面程序中，分别输入了不同系数的数值，求出不同类型的方程的解，这样做可以检测程序中是否具有各种情况下的适应性。

最后，在使用分支结构时，需要注意以下的几个问题。

- 分支结构是程序结构中最重要的结构之一，可以使用任意多个elseif语句，但是只能有一个if语句和一个end语句。
- 分支语句可以相互嵌套，可以根据实际需要将各个分支语句进行嵌套，以解决较复杂的问题。

7.4.3 switch 分支结构

和前面介绍的switch分支结构类似，在MATLAB中通过判断条件表达式的结果，选择一个数控的多个开关。其一般的语法调用方式如下：

```
switch expression (scalar or string)
   case value1
      statements
   case value2
      statements
   ..................
   otherwise
      statements
end
```

在上面的语法结构中，expression是一个标量或者字符串，MATLAB会将表达式的数值结果和每个case后面的数值进行比较，如果比较结果为真，MATLAB会取下一个数值来进行比较，直到比较结果为真，MATLAB执行相应的命令，然后跳出该分支结构。如果所有的比较结果都为假，也就是表达式的数值和所有的检测值都不匹配，MATLAB执行otherwise部分的语句。

例7.15 在MATLAB中，使用switch分支结构来判断输入A的数值。

Step 1 单击MATLAB命令窗口工具栏中的D按钮，打开M文件编辑器，在M文件编辑器中输入下面的程序代码

```
%提示用户输入数值
input_num=input('Enter the number:');
%根据情况判断数值大小，显示数值信息
switch input_num
    case -1
        disp('negative one');
    case 0
        disp('zero');
    case 1
        disp('positive one');
        %如果不是上面的数值，显示'其他数值'
    otherwise
        disp('other value');
end
```

step 2 单击MATLAB编辑器中的"保存"按钮，将[illegible]，保存结果如图7.20所示。

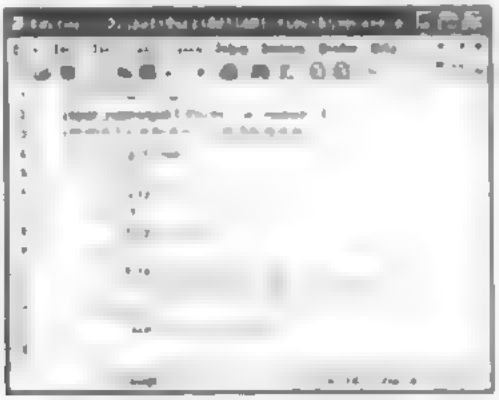


图7.20 保存程序代码

step 3 在MATLAB命令窗口中，输入"swcase"，并按"Enter"键，程序开始运行。依次输入不同的数值，得到的结果如下

```
>> swcase
Enter the number:1
positive one
>> swcase
Enter the number:-1
negative one
>> swcase
Enter the number:0
zero
>> swcase
Enter the number:8
other value
```

在switch分支结构中，case命令后的检测不仅可以为一个标量或者字符串，还可以为一个元胞数组，如果检测值是一个元胞数组，MATLAB会将表达式的值和元胞数组中的所有元素进行比较，如果元胞数组中某个元素与表达式的值相等，MATLAB，则比较结果为真。

7.4.4 try-catch 结构

在 MATLAB 中，try-catch 结构为这种私error变化，主要用来对异常情况进行处理。其相应的语法结构如下

```
try
    statement
    ...
    statement
catch
    statement
    ...
    statement
end
```

在上面的语法结构中，try 后面的命令语句会被执行，只有当这些语句执行出现错误时，catch 控制语句就会捕获之，执行相应的语句。如果执行 catch 语句后的命令又出现错误，MATLAB 就会终止该程序结构。

例 7.16 在 MATLAB 中，使用 try-catch 语法结构来捕获错误。

Step 1 单击 MATLAB 命令窗口工具栏中的 按钮，打开 M 文件编辑器。在 M 文件编辑器中输入下面的程序代码

```
% 初始化数组
a=[ 1 -3 2 5];
try
% 显示某个元素
    index=input('Enter subscript of element to display: ');
    disp([ 'a(' int2str(index) ') = ' num2str(a(index))] );
catch
  % 如果发现出错
  disp([ 'Illegal subscript: ' int2str(index)] );
  % 显示错误类型
  A=lasterr;
  disp([ 'The type of error: ' A ]);
end
```

Step 2 单击 M 文件编辑器中的“保存”按钮，将上面的程序代码保存为“calc_root.m”，得到的结果如图 7.21 所示。

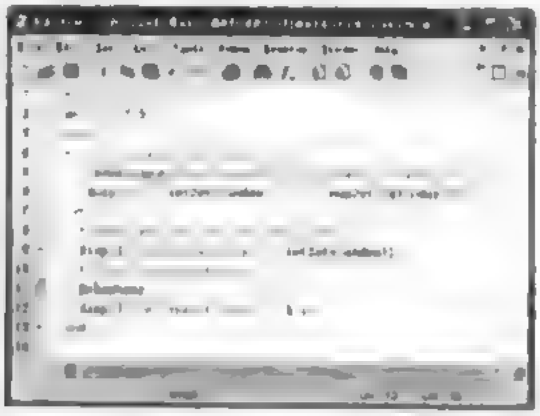


图 7.21 保存程序代码

step 3 在 MATLAB 的命令窗口中，输入 try_catch，然后按下"Enter"键，[illegible]，依据不同的数值情况，得到的结果如下

```
>> try_catch
Enter subscript of element to display: 2
a(2) = -3
>> try_catch
Enter subscript of element to display: 6
Illegal subscript: 6
The type of error: Index exceeds matrix dimensions.
```

从以上结果可以看出，当输入的数组下标超过了数组的维数时，MATLAB 会给出错误提示，同时给出系统的错误信息"Index exceeds matrix dimensions"。

在 MATLAB 中，lasterr 命令[illegible]，该命令[illegible]字符串[illegible] try catch [illegible]。[illegible] MATLAB 中的[illegible]。

7.4.5 while 循环结构

[illegible] MATLAB [illegible]循环结构[illegible]，循环[illegible]条件。

在 MATLAB 中，while 循环结构的语法形式如下

```
while expression
    statements
end
```

while 循环结构中的表达式[illegible]值为真，循环体部分将执行[illegible]，[illegible]数组，则 while 循环要求所有的元素都为真。

例 7.17 在 MATLAB 中，使用 while 循环结构对输入进行统计，计算输入数值数组的平均值和标准差。

step 1 分析[illegible]平均值和标准差的计算公式分别如下

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$s = \sqrt{\frac{N\sum_{i=1}^{N} x_i^2 - \left(\sum_{i=1}^{N} x_i\right)^2}{N(N-1)}}$$

其中，参数 N 表示用户输入的数值个数，x 表示每个输入的数值。[illegible]。MATLAB 提供 mean 和 std 函数来计算[illegible]，为了演示 while 循环结构，编写计算这些参量的简单代码。

step 2 单击 MATLAB 命令窗口工具栏中的□按钮，打开 M 文件编辑器。在 M 文件编辑器中输入下面

保存代码

```
% Script file stats.m
%
% Purpose:
%   To calculate mean and the standard deviation of
% an input data set containing and arbitrary number
% of input values.
%
% Define variables:
%  n        The number of input samples
% std_dev   The standard devation of the input samples
% sum1      The sum of the input values
% sum2      The sum of the squares of the input values
% x         input data value
% xvar      The average of the input samples
% Initialize variables
n=0;sum1=0;sum2=0;
%Read the input values
x=input('Enter the first value: ');

%创建未知数组元素和的循环结构
while x~=0
    n=n+1;
    sum1=sum1+x;
    sum2=sum2+x^2;

    % 读入原始数据
    x=input('Enter next value: ');
end
%计算平均值和标准方差
xvar=sum1/n;
std_dev=sqrt(1/(n*(n-1))*(n*sum2-sum1^2));
%在命令窗口显示结果
fprintf('The mean of this data set is: %f\n',xvar);
fprintf('The standard deviation is: %f\n',std_dev);
fprintf('The number of data is: %d\n',n);
```

Step 3 单击MATLAB文件编辑器工具栏的“保存”按钮，将上面的程序代码保存为“stats.m”，保存的结果如图7.22所示。

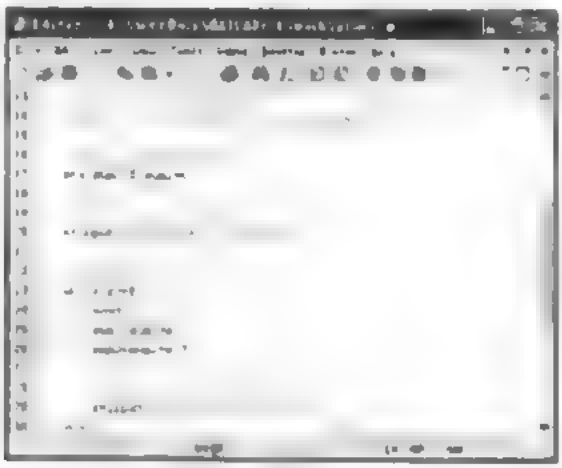


图7.22 保存程序代码

step 4 返回MATLAB的命令窗口，输入“stats”，然后按“Enter”键，根据程序代码的提示，依据不同的数值情况，得到的结果如下：

```
>> stats
Enter the first value: 6
Enter next value: 7
Enter next value: 9
Enter next value: 12
Enter next value: 0
Enter next value: 5
Enter next value: -7
The mean of this data set is: 6.500000
The standard deviation is: 4.037326
The number of data is: 6
```

从上面的实例可以看出，当输入任意的负数时，MATLAB会自动跳出while循环结构，只统计前面输入非负数值的统计量，得到相应的统计结果。

step 5 使用MATLAB的内置函数进行数值统计，检测程序是否正确，得到的结果如下：

```
>> A=[ 6,7,9,12,0,5] ;
>>mean_A=mean(A,2);
>>std_A=std(A,0,2);
>> size_A=size(A,2);
>> mean_A
mean_A =
    6.5000
>> std_A
std A =
    4.0373
>> size_A
size A =
     6
```

从上面的结果中可以看出，使用MATLAB内置的函数进行数据量的统计时，得到的结果和上面编写的程序结果相同。

for循环结构

在MATLAB中，另外一种常见的循环结构是for循环结构，其常用的调用格式如下：

```
for variable = expression
    statements
end
```

在上面的语法结构中，variable称为循环变量，循环体被重复执行的次数是确定的，该次数由for命令后的表达式expression决定。

例7.18 在MATLAB中使用for循环结构完成上面小节的数值统计功能。

step 1 单击MATLAB命令窗口工具栏中的□按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
% Script file stats2.m
```

```
%
% Purpose:
%   To calculate mean and the standard deviation of
%  an input data set containing and arbitrary number
%  of input values.
%
% Define variables:
%  n        The number of input samples
% std_dev   The standard devation of the input samples
% sum1      The sum of the input values
% sum2      The sum of the squares of the input values
% x         Input data values
% xvar      The average of the input samples
% Initalize variables
sum1=0;sum2=0;

% 输入排序数据的个数
n=input('Enter the number of points: ');
% Check to see if we have enough input data.
if n<2
    disp('At least 2 values must be entered.');
else
    % 创建计算总和和平方和的循环
    for ii=1:n
        x=input('Enter value: ');
        sum1=sum1+x;
        sum2=sum2+x^2;
    end
%计算平均值和标准方差
xvar=sum1/n;
std_dev=sqrt((n*sum2-sum1^2)/(n*(n-1)));
%Print the results
fprintf('The mean of this data set is: %f\n',xvar);
fprintf('The standard deviation is: %f\n',std_dev);
fprintf('The number of data is: %d\n',n);
end
```

Step 2 单击自定义编辑器中的“保存”按钮，将上面的程序保存为“stats2.m”，保存结果如图 7.23 所示。

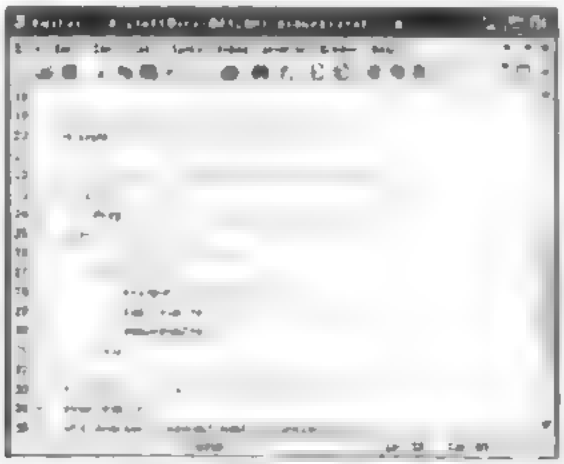


图 7.23 保存程序代码

step 3 返回到MATLAB的命令窗口，输入"stats2"，然后按"Enter"键，根据程序代码的提示，依据不同的数值情况，得到的结果如下

```
>> stats2
Enter the number of points: 7
Enter value: 1
Enter value: 6
Enter value: -7
Enter value: 0
Enter value: -2
Enter value: 3
Enter value: 4
The mean of this data set is: 1.285714
The standard deviation is: 4.609481
The number of data is: 7
```

在上面的程序代码中，使用for循环替代了前面小节中的while循环结构，同样得出数值的统计结果，不过，这段程序代码通过数值个数来控制循环，可以统计的数值顶多达到255，可以统计包括负数在内的统计信息。

while语句和for循环都是比较常用的循环结构，但是两个循环结构有显著区别的其中最明显的，则在于，while循环的执行次数是不确定的，而for循环的执行次数是确定的。

7.4.7 综合实例

前面经验的基础上，对于比较复杂的MATLAB程序，还需要将前面章节介绍的程序结构综合起来使用，才能解决复杂的问题。

例7.19 在MATLAB中，通过程序来演示小球的抛物线轨迹。

step 1 分析小球的抛物线轨迹模型。假定在初始时刻的速度，也就是小球的初始速度是v_0，以及小球的初始角度是θ。根据基础的物理知识，小球在水平和垂直方向上的速度分量分别为

$$\begin{cases} v_{0x} = v_0 \cos\theta \\ v_{0y} = v_0 \sin\theta \end{cases}$$

在本实例中，程序代码要求解的是抛物线轨迹上水平距离的最大距离。根据相关知识，其距离的求解公式如下

$$\begin{cases} t = -\dfrac{2v_{0y}}{g} \\ x_{max} = v_{0x} t \end{cases}$$

在上面的公式中，g代表的是垂直方向速度，在本实例中参数选择的数值为-9.82。当然它的。小球在垂直方向上的最高距离为：

$$y_{max} = \frac{v_{0y}^2}{2g}$$

step 2 根据本实例的要求，可以输入抛射小球的初始速度，然后得出相应的计算数据。单击MATLAB命令窗口工具栏中的□按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
% Script file ball.m
%
% Purpose:
%    This program calculates the distance traveled by a ball
%  thrown at a specified angle "theta" and a specified velocity
%  "vo" form a point,ignoring air friction .It calculates the angle
%  yeilding maximum range,and also plots selected trajectories.
%
% Define variables:
%   conv       degrees to radians conv factor
%   grav       The gravity accel
%   ii,jj      Loop index
%   index      The maximum range in array
%   maxangle   The angle that gives the maximum range
%   maxrange   Maximum range
%   range      range for a specified angle
%   time       Time
%   theta      Initial angle
%   fly_time   the total trajectory time
%   vo         The initial velocity
%   vxo        x-component of the initial velocity
%   vyo        y-component of the initial velocity
%   x          x-position of ball
%   y          y-position of ball

% 定义常数数值
conv=pi/180;
grav=-9.82;
vo=input('Enter the initial velocity:');

range=zeros(1,91);

% 计算最大的水平距离
for ii=1:91
     theta=ii-1;
     vxo=vo*cos(theta*conv);
     vyo=vo*sin(theta*conv);
     max_time=-2*vyo/grav;
     range(ii)=vxo*max_time;
end

%显示计算水平距离的列表
fprintf('Range versus angle theta:\n');
for ii=1:5:91
    theta=ii-1;
    fprintf('%2d %8.4f\n',theta,range(ii));
end

% 计算最大的角度和水平距离
[maxrange index]=max(range);
```

```
maxangle=index 1;
fprintf('\n Max range is %8.4f at %2d degrees.\n',maxrange,maxangle);

% 绘制轨迹图形
for ii=5:10:80
     theta=ii;
     vxo=vo*cos(theta*conv);
     vyo=vo*sin(theta*conv);
     max_time=-2*vyo/grav;
     %  计算小球轨迹的x，y坐标数值
     x=zeros(1,21);
     y=zeros(1,21);
     for jj=1:21
         time=(jj-1)*max_time/20;
         x(jj)=vxo*time;
         y(jj)=vyo*time+0.5*grav*time^2;
     end
     plot(x,y,'g');
     if ii==5
         hold on;
     end
end
% 添加图形的标题和坐标轴名称
title('\bf Trajectory of Ball vs Initial Angle\theta');
xlabel('\bf\itx \rm\bf(meters)');
ylabel('\bf\ity \rm\bf(meters)');
axis([ 0 max(range)+5 0 -vo^2/2/grav] );
grid on;

% 绘制最大水平的轨迹图形
vxo=vo*cos(maxangle*conv);
vyo=vo*sin(maxangle*conv);
max_time=-2*vyo/grav;
     %  Calculate the (x,y) position
     x=zeros(1,21);
     y=zeros(1,21);
     for jj=1:21
         time=(jj-1)*max_time/20;
         x(jj)=vxo*time;
         y(jj)=vyo*time+0.5*grav*time^2;
     end
     plot(x,y,'r','Linewidth',2);
     hold off
```

step 3 单击M文件编辑器中的"保存"按钮，将上面的程序代码保存为"ball.m"，得到的结果如图7.24所示。

step 4 返回到MATLAB的命令窗口，输入"ball"，然后按"Enter"键，根据程序代码的提示，依据不同的数值情况，得到的结果如下：

```
>> ball
Enter the initial velocity:20
Range versus angle theta:
 0    0.0000
 5    7.0732
```

```
 10   13.9316
 15   20.3666
 20   26.1828
 25   31.2034
 30   35.2760
 35   38.2767
 40   40.1144
 45   40.7332
 50   40.1144
 55   38.2767
 60   35.2760
 65   31.2034
 70   26.1828
 75   20.3666
 80   13.9316
 85    7.0732
 90    0.0000
  Max range is  40.7332 at 45 degrees.
```

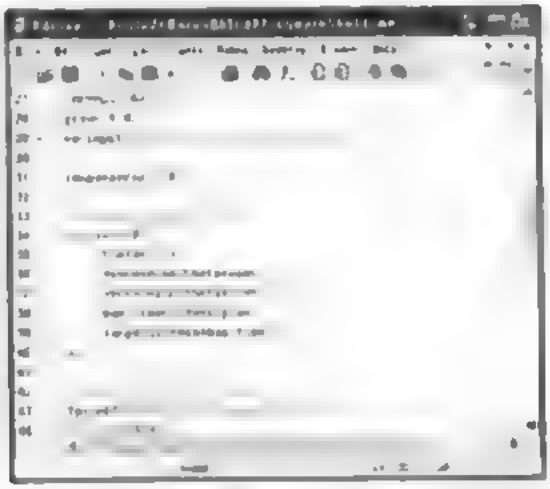


图 7.24　保存程序代码

除了上面的数值结果之外，MATLAB 还会绘制相应的运动轨迹，如图 7.25 所示。

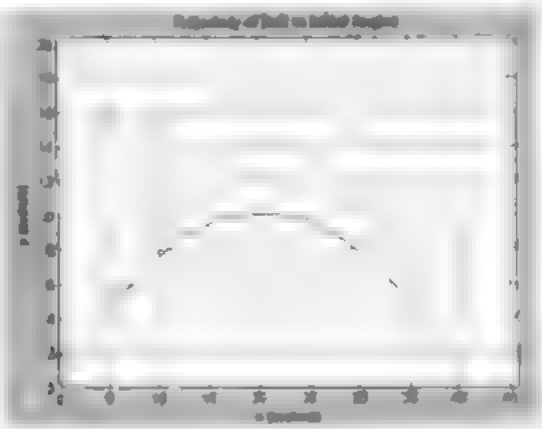


图 7.25　初始速度为 20 时的轨迹

Step 5 修改初始速度数值，将其改为 45，得到的结果如下

```
>> ball
```

```
Enter the initial velocity:45
Range versus angle theta:
 0   0.0000
 5  35.8083
10  70.5286
15 103.1059
20 132.5504
25 157.9674
30 178.5847
35 193.7757
40 203.0790
45 206.2118
50 203.0790
55 193.7757
60 178.5847
65 157.9674
70 132.5504
75 103.1059
80  70.5286
85  35.8083
90   0.0000
 Max range is 206.2118 at 45 degrees
```

同时，MATLAB会给出对应的图形结果，如图7.26所示。

图7.26 初始速度为45时的轨迹

[illegible]

7.5 MATLAB的控制语句

在使用MATLAB进行编程时，经常遇到需要[illegible]情况，则[illegible]对于错误信息等情况，则还需要[illegible]。在MATLAB中，对应的控制语句有continue、break、return、error等。

7.5.1 continue 命令

在 MATLAB 中，continue 命令功能是结束指令的执行，但是跳过循环体中剩余在执行的语句，其使用格式与 break 相同，直接在指令中输入 continue 即可。下面使用一个简单的实例，说明 continue 命令的使用方法。

例 7.20 使用简单的实例说明 continue 命令的使用方法。

step 1 单击 MATLAB 的命令窗口工具栏中的按钮，打开 M 文件编辑器。在 M 文件编辑器中输入一段的程序代码

```
for ii=1:9
    if ii==3
        continue
    end
    fprintf('ii=%d\n',ii);
    if ii==5
        break
    end
end
disp('The end of loop')
```

在前面的程序代码中增加了代码的复杂度，去掉了所有的注释，不影响程序的运行，在后面的程序中将注释符号删除后都可以重新运行了。

step 2 将上面的代码保存为"break_continue.m"文件，在 MATLAB 的命令窗口中输入"break_continue"，然后按"Enter"键，则可以得到如下的结果如下

```
>> break_continue
ii=1
ii=2
ii=4
ii=5
ii=6
ii=7
ii=8
ii=9
The end of loop
```

step 3 打开"break_contine.m"文件，在编辑器中修改其代码，得到的结果如下

```
for ii=1:9
    if ii==4
        continue
    end
    fprintf('ii=%d\n',ii);
   if ii==5
        break
        end
   end
disp('The end of loop')
```

上面程序的结构与上一步的前面的主程序的结构，只是保存这种结构的顺序。

step 6 在 MATLAB的命令窗口中输入"break_continue"，然后按"Enter"键，就可以调用该程序了，结果如下

```
>> break_continue
i=1
i=2
i=4
i=5
The end of loop
```

在上面程序代码中使用了break语句，其功能是跳出当前的程序代码，请在下面的章节中进行详细介绍。

7.5.2 break 命令

在 MATLAB 中，break 命令常用在循环体，表示循环，即中断当前循环。一个 for 或者 while 循环的结束，也常用在条件语句中，需要与 if 语句结合起来应用在循环体中。

例 7.21 在 MATLAB 中求 Fibonacci 数列的第一个大于 200 的项，并输出该数列的长度。

step 1 在 MATLAB 命令窗口，选择菜单命令"File"，打开 M 文件编辑器，在 M 文件编辑器中输入下面的程序代码

```
i=[1,1,zeros(1,n-2)];
for i=3:n,
    a(i)=a(i-1)+a(i-2);
    if a(i)>=200
        a(i)
        break;
    end
end
i
```

step 2 将上面程序代码保存为"fib.m"文件，然后在 MATLAB 命令窗口中输入"fib"，按"Enter"键，就可以得到对应的结果如下

```
>> fib
ans =
   987
i =
    16
```

从上面程序运行结果中可以看出，在 Fibonacci 数列中第一个大于 200 的数值是 987，其对应的数列长度是 16。

7.5.3 return 命令

在通常情况下，当调用函数执行完毕后，MATLAB 会自动返回主调函数或者键盘。如果在被调函数中插入 return 命令后，可以强制 MATLAB 结束当前的函数并返回调用。

return 命令可使正在执行的函数立即结束，即中断调用函数的执行，经常用于函数的末尾用来正常结束函数的运行。

在MATLAB的大量函数中，很多函数在标准代码中，大多数有return命令，在此中一个简单的det函数代码如下

```
function d = det(A)
%DET det(A) is the determinant of A.
if isempty(A)
    d = 1;
    return
else
    ...
end
```

在上面的标准代码中，首先通过函数调用语句来判断参数A的类型，当A是空数组时，直接返回值=1，然后结束程序代码。

7.5.4 input命令

在MATLAB中，input命令主要功能是将MATLAB的控制权暂时交给用户，然后，由用户通过键盘输入数值、字符串或者表达式，通过按"Enter"键将输入的内容输入到工作空间中，后面将控制权交回MATLAB。其常用的调用格式如下

- user_entry = input('prompt')：将用户键入的内容赋给变量，即user_entry
- user_entry = input('prompt','s')：将用户键入的字符串赋给变量user_entry

对于上面的第一个调用格式，可以输入数值、字符串、向量或矩阵等各种类型的数据。第二个调用格式，只接收字符串的变量，若分号字符串的单元数组变量user_entry

例7.22 在MATLAB中演示如何使用input函数。

step 1 单击MATLAB主窗口工具栏中的新建按钮，打开M文件编辑器，在M文件编辑器输入下面的程序代码

```
function test_input()
reply = input('Do you want more? Y/N [Y]: ','s');
if isempty(reply)
    reply = 'Y';
end
if reply=='Y'
    disp('you have selected more information');
else
    disp('you have selected the end');
end
```

step 2 将上面的代码保存为"test_input.m"文件，然后在MATLAB的命令窗口中输入"test_input"，按"Enter"键，就可以得到对应的结果如下

```
>> test_input
Do you want more? Y/N [Y]:
you have selected more information
>> test_input
```

```
Do you want more? Y/N [Y]: N
you have selected the end
>> test_input
Do you want more? Y/N [Y]: Y
you have selected more information
```

[illegible]

7.5.5 keyboard 命令

在 MATLAB 中，将 keyboard 命令放置在 M 文件中，当使用 [illegible] 键盘，[illegible] 在 M 文件中使用该命令，[illegible]

例 7.23 在 MATLAB 中，演示如何使用 keyboard 命令。

在 MATLAB 的命令窗口中输入下面的内容

```
>> keyboard
K>>  for ii=1:9
     if ii==3
          [illegible]
     end
     [illegible]
    if ii==5
          kreak
          [illegible]
    end
ii=1
[illegible]
ii=4
[illegible]
K>> return
[illegible]
```

从上面的程序执行中可以看出，当输入 keyboard 命令后，在每一行的开始前会显示一个提示符，而当用户输入 return 后，提示符恢复正常的提示效果。

在 MATLAB 中，keyboard 命令和 input 命令[illegible]命令[illegible]输入[illegible]个 MATLAB 命令，[illegible]

7.5.6 error 和 warning 命令

在 MATLAB 中，编写 M 文件的[illegible]，MATLAB 都提供了[illegible]两个有用的命令。

- error('message') 显示出错信息 message，终止程序。
- [illegible]

◆ warning('message')：显示警告信息 message，程序继续进行。

例 7.24 借由前面一节中的 "stats2.m"，修改其部分程序代码，使用不同的警告样式，查看 MATLAB 的不同错误提示模式。

Step 1 在 MATLAB 中，打开 "stats2.m" 文件，在编辑器中修改其程序代码，并将其保存为 "error_message.m" 文件，修改的程序代码如下

```
sum1=0;sum2=0;
% Get the number of points to input
n=input('Enter the number of points: ');
% Check to see if we have enough input data.
if n<2
    error('Not enough input data','Data Error');
else
    ...............................................
...其余程序与其他部分，在这里全部省略
end
```

Step 2 单击 MATLAB 的命令窗口，在命令窗口中输入 "error_message"，然后输入数值 1，程序的结果如图 7.27 所示。

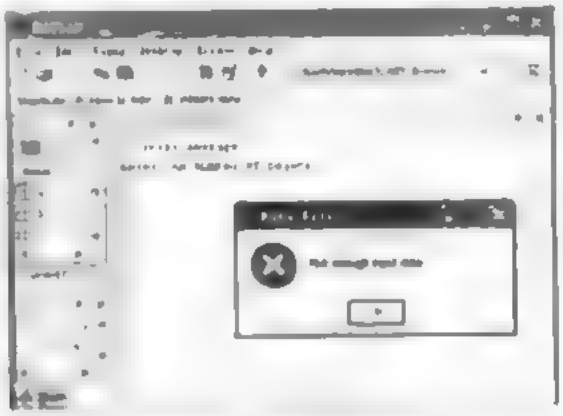


图 7.27 显示错误信息

当用户输入的数值个数小于 2 时，MATLAB 调用错误信息对话框。当用户单击对话框中的 "OK" 按钮后，将自动退出程序代码。

Step 3 打开 "error_message.m" 文件，在编辑器中修改其程序代码，然后保存其已经修改的程序代码，修改的程序代码如下

```
sum1=0;sum2=0;
% Get the number of points to input
n=input('Enter the number of points: ');
% Check to see if we have enough input data.
if n<2
    error('Not enough input data');
else
    ...............................................
end
```

Step 4 单击 MATLAB 命令窗口，在命令窗口中输入 "error_message"，然后输入数值 1，程序的结

果如下

```
>> error_message
Enter the number of points: 1
??? Error using ==> error_message
Not enough input data
Error in ==> error_message at 24
    error('Not enough input data');
>>
```

Step 3 打开"error_message.m"文件，在编辑器中修改其程序代码，然后保存修改的程序代码，修改的程序代码如下

```
num1=0;num2=0;
% Get the number of points to input
n=input('Enter the number of points: ');
% Check to see if we have enough input data.
if n<2
    warning('Not enough input data');
else
    ...... ..... ..... ..... ..... ...
    end
```

Step 5 进入MATLAB命令窗口，在命令窗口中输入"error_message"，并输入数值1，观察运行结果如下

```
>> error_message
Enter the number of points: 1
Warning: Not enough input data
> In error_message at 24
```

在上面的程序代码中，演示了MATLAB中的不同错误信息输出方式，其中error和warning的主要区别在于warning命令指示警告信息后继续运行程序。

7.6 程序的向量化概念

在MATLAB中，除了前面介绍的三大程序结构之外，还有一种特殊的方法——程序的向量化，向量化是一个程序概念，指的是使用向量化的程序代码来替代循环结构，使用向量化可以MATLAB的程序性能带来质的提高。

在本节中，将介绍如何使用逻辑数组等向量化方法来替代MATLAB中的循环结构，提高程序的性能。

7.6.1 程序的向量化

在不使用向量化的情况下，使用一个简单的案例来对比循环结构和向量结构的差异，该案例的目的是，计算数组元素的平方，并分别记录两种方法的运行时间，数组元素是从1到1万的整数，实现该功能的方法可以是循环结构，也可以是向量化程序，下面详细介绍这两种方法。

例 7.25 在 MATLAB 中，使用循环结构和向量化来求解数组的乘幂计算。

step 1 单击 MATLAB 命令窗口工具栏中的 D 按钮，打开 M 文件编辑器。在 M 文件编辑器中输入下面的程序代码：

```
for ii=1:100
square(ii)=ii^2;
square_root(ii)=ii^(1/2);
cube_root(ii)=ii^(1/3);
end
disp('Square   Square_root  Cube_root')
result=[ square',square_root',cube_root'];
```

step 2 将上面的代码保存为 "cycle.m" 文件。然后单击 MATLAB 命令窗口工具栏中的 D 按钮，打开一个新的 M 文件编辑器。在 M 文件编辑器中输入下面的程序代码：

```
ii=1:100;
square=ii.^2;
square_root=ii.^(1/2);
cube_root=ii.^(1/3);
disp('Square   Square_root  Cube_root')
[ square',square_root',cube_root']
```

step 3 将上面的代码保存为 "vector.m" 文件。然后返回 MATLAB 的命令窗口，在命令窗口中输入 "cycle"，然后按 "Enter" 键，得到的结果如下：

```
>> format short g
>> cycle
Square   Square root  Cube root
result =
            1            1            1
            4       1.4142       1.2599
            9       1.7321       1.4422
           16            2       1.5874
           25       2.2361         1.71
           36       2.4495       1.8171
           ..........................................
         4225       8.0623       4.0207
         4356        8.124       4.0412
         4489       8.1854       4.0615
         4624       8.2462       4.0817
         4761       8.3066       4.1016
         ...............................................
         9409       9.8489       4.5947
         9604       9.8995       4.6104
         9801       9.9499       4.6261
        10000           10       4.6416
```

step 4 在 MATLAB 的命令窗口中输入 "vector"，查看向量化程序的结果：

```
>> vector
Square   Square_root  Cube_root
ans =
            1            1            1
```

4	1.4142	1.2599
[illegible]	1.7321	1.4422
16	2	1.5874
25	2.2361	1.71
36	2.4495	1.8171
[illegible]	[illegible]	[illegible]
4225	8.0623	4.0207
4356	8.124	4.0412
[illegible]	[illegible]	[illegible]
4624	8.2462	4.0817
4761	8.3066	4.1016
[illegible]	[illegible]	[illegible]
9409	9.8489	4.5947
9604	9.8995	4.6104
9801	9.9499	4.6261
10000	10	4.6416

[illegible]

[illegible]

7.6.2 向量化和循环结构对比

[illegible]

例 7.26 [illegible]

Step 1 [illegible]

```
% Script file timings.m
%
%  目的:
%  [illegible]
%  [illegible]时间
%  1.不使用初始化数组的数值
%  2.使用初始化数组数值
%  3.使用向量化

% 定义变量
% ii,jj        循环次数
```

```
% ave1        第一种方法的平均时间
% ave2        第二种方法的平均时间
% ave3        第三种方法的平均时间
% maxcount    循环计算的次数
% square      平方数值的数组

% 不预先定义数组，直接求解结果
[illegible]
[illegible]
for jj=1:maxcount
clear square
[illegible]
    [illegible]
end
end
[illegible];

% 预先定义空数组，计算循环
maxcount=50;
[illegible];
[illegible]
clear square
square=zeros(1,10000);
for ii=1:10000
    square(ii)=ii^2;
end
end
[illegible]

% 使用程序的向量化
maxcount=100;
tic;
for jj=1:maxcount
[illegible]
    [illegible];
    square=ii.^2;
[illegible]
[illegible]
%display the results
fprintf('Loop/uninitialized array= %9.5f\n',ave1);
[illegible]
[illegible]
```

从上面程序的结果，我们可以看出，第一种方法的平均时间最长，第二种方法的平均时间稍有减少，而第三种方法的平均时间则明显减少了。

step 2 将上面的程序保存为"timing.m"文件，在MATLAB命令窗口中输入"timing"，然后按"Enter"键，就可以得到对应的结果。

```
>> timing
Loop/uninitialized array=   0.47100
loop/initialized array=   0.00020
vectorized=   0.00010
```

[illegible]

7.6.3 逻辑数组

[illegible]

例7.27 在MATLAB中创建逻辑数组，并通过命令查看逻辑数组

step 1 在MATLAB的命令窗口中输入下面的内容

```
>> a=[1,2,3;4,5,6;7,8,9];
>> b=a>4;
```

step 2 查看上面步骤保存的数组b，然后[illegible]

```
>> b
b =
     0     0     0
     0     0     1
     1     1     1
>> whos
  Name      Size                    Bytes  Class
  a         3x3                        72  double array
  b         3x3                         9  logical array
Grand total is 18 elements using 81 bytes
```

从上面的结果中可以看出，数组b[illegible]，而a是[illegible]，b是逻辑数组。

7.6.4 使用logical命令创建逻辑数组

除了使用关系表达式和逻辑表达式之外，还可以使用logical命令创建逻辑数组。这种使用命令可以实现很多逻辑功能。

例7.28 在MATLAB中使用logical命令创建逻辑数组，并进行数组运算

step 1 在MATLAB主命令窗口中输入下面的内容

```
>> a=[1,2,3;4,5,6;7,8,9];
>>b=logical(eye(3));
>> b=a>5;
>> b=a>5;
```

step 2 在命令窗口中继续输入一个[illegible]

```
>> b
b =
     1     0     0
     0     1     0
     0     0     1
>> c
c =
     1
     5
     9
>> d
d =
     1     2     3
     4     5     6
     7     8    10
```

step 3 查看各个数组的数据类型，得到的结果如下：

```
>> whos
  Name      Size                    Bytes  Class
  a         3x3                        72  double array
  b         3x3                         9  logical array
  c         3x1                        24  double array
  d         3x3                        72  double array
Grand total is 30 elements using 177 bytes
```

[illegible]

7.6.5 逻辑数组和向量化

[illegible]

例 7.29 [illegible]

step 1 [illegible]

```
% Script file logical.m

% 目的:
%     [illegible]
%   使用下面两种不同的方法
%   1.使用循环和 if 结构
%   2.使用逻辑数组

% 工作变量
% i,j,k      循环变量
% eval       第一种方法的平均时间
```

```
% ave2      第二种方法的平均时间
% maxcount 循环结算的时间次数

%计算循环结构
max[illegible]=1;
[illegible]
[illegible] max [illegible]
    a=1:10000;
    [illegible]
        if a(ii)>6000
            [illegible];
        end
    end
end

%使用逻辑数组的方法计算
ave1=(toc)/maxcount;
max[illegible]=1;
[illegible]
[illegible]
    a=[illegible];
    b=a>6000;
    a(b)=sqrt(a(b));
end
a[illegible]=([illegible] max [illegible]);

%显示结果
fprintf('Loop if approach= %9.5f\n',ave1);
[illegible]
```

step 2 将上述程序保存为 "logical.m" 文件，在MATLAB的命令窗口中输入 "logical"，然后按 "Enter" 键，就可以得到对应的结果如下

```
>> logical
Loop if approach=   0.02100
Logical array approach=   0.00100
```

上面的程序[illegible]是MATLAB[illegible]，[illegible]，[illegible]，[illegible]，[illegible]

在本节的[illegible]中，为了[illegible]，[illegible]，[illegible]，[illegible]

7.7 脚本和函数

前面[illegible]，在MATLAB的编辑器中可以建立脚本文件和函数文件。这两种类型的文件在使用方面都很相似，但是在语法和使用上还是有很多不同之处的。在7.1节的第[illegible]等等的实例，编辑过脚本文件和函数文件，请参照[illegible]。本节中将介绍脚本文件。

7.7.1 脚本文件

[illegible]

相对于函数文件，脚本文件的构成比较简单，主要特点如下。

- 脚本文件是一串按用户意愿排列而成的MATLAB命令窗口。
- [illegible]

7.7.2 脚本文件实例

例7.30 在MATLAB中，编写绘制三维锥空图形的脚本文件。

解答 [illegible]

```
% 脚本文件 loukong.m
[illegible]
%旋转圆柱
[illegible]
%产生旋转柱面的数据
[x,y,z]=cylinder(r,60);
%确定要应用的数据下标
zi=find(x<0&y<0);
%进行更改剪切
[illegible]
surf(x,y,z);
colormap(hsv),shading interp
%设置光源
[illegible]
%高亮度反射
material([0.5,0.4,0.3,10,0.3])
```

解答 [illegible] "Enter"键，得到的图形如图7.28所示。

解答 [illegible]

[illegible]

图7.28 程序代码运行的结果

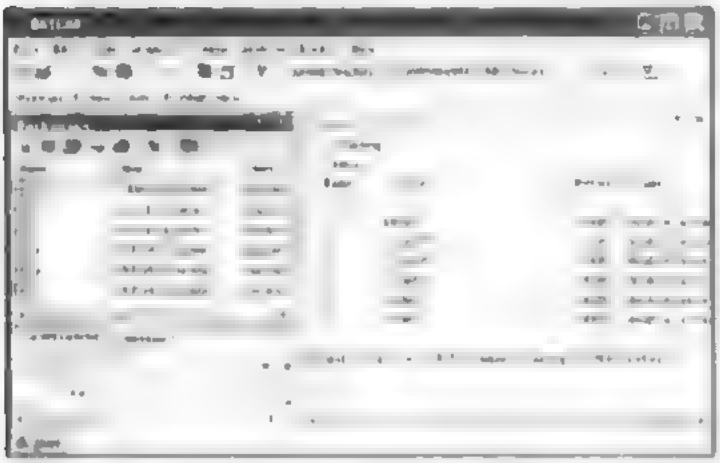


图7.29 查看工作空间的变量

7.7.3 函数文件

[illegible]

- [illegible]，也可以输入任意数目的变量。
- MATLAB会在输入[illegible]数目的变量，[illegible]函数可调出。
- [illegible]的中间变量都会被清除。
- 函数[illegible]的M函数文件的[illegible]，随[illegible]相对的。[illegible]MATLAB[illegible]任意多个M函数文件。
- 如果存函数文件[illegible]，而不是存放在基本空间中。

7.7.4 函数文件实例

例7.31 在MATLAB中，编[illegible]函数，其中[illegible]函数[illegible]

step 1 单击MATLAB菜单栏上的[新建]→[M文件]选项，打开M文件编辑器，在M文件编辑器中输入下面的程序代码。

```
function f=myfun(x)
y1=-x.^2+4*x+450;
y2=[illegible]*x.
f=y1+y2;
```

step 2 保存函数文件为"myfun.m"，进入MATLAB命令窗口，输入下面的命令。

```
>> fh=@myfun;
>> ezplot(fh,[0,35]);
>> grid
```

step 3 查看命令执行结果，当输入上面的命令后，按"Enter"键，则会显示如下图。

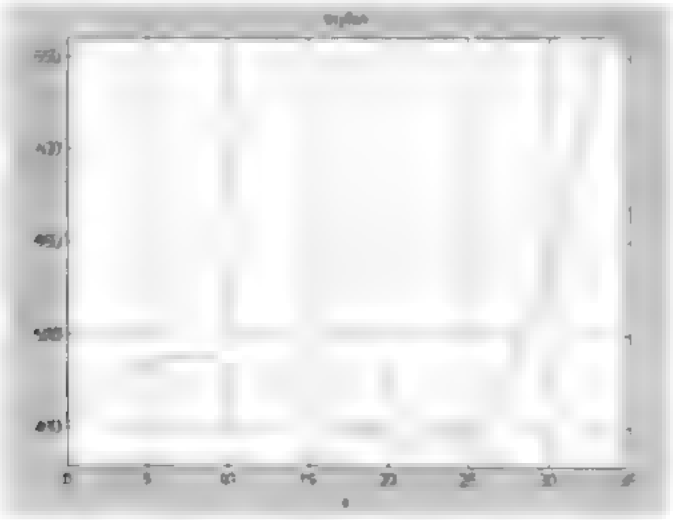


图7.30 绘制的结果

step 4 在MATLAB命令窗口，可查看工作区内的变量信息，则会显示如下：

```
>> whos
  Name      Size                    Bytes  Class
  fh        1x1                        16  function_handle array
Grand total is 1 element using 16 bytes
```

从上面的结果可以看出，函数产生一个变量fh，它是一个函数句柄，工作区中只有一个变量fh。

[illegible]

7.7.5 P码文件

在MATLAB中，一个M文件一般表示一个函数，但一般情况下，不在MATLAB的命令窗

口中输入文件名，还是使用M文件编辑器打开对应文件，MATLAB都会对该M文件进行语法分析，并把生成的相应内部伪代码存放在内存中。这个内部伪代码简称P码，全称为Psedocode。当再次调用该M文件时，MATLAB会直接运行该文件在内存中的P码文件而不会对原始代码进行重复分析。同时，在MATLAB中，分析器总是把M文件连同被它调用的所有函数M文件一起被变换为P码文件。

P码文件和源代码文件有相同的文件名，扩展名为.p。其运行速度要高于源代码文件，这种速度优势在规模较大的文件中体现得更加明显。

除了MATLAB自动调用之外，可以使用相应的命令来创建P码文件。创建P码文件的主要功能在于保存密码，二进制的P码文件可以执行代码的功能，但是不易于阅读，可以起到很好的保密效果。

创建P码文件的常用命令如下：

```
pcode filename                  在当前目录下创建 filename.p
pcode. filename -inplace        在 filename.m 目录下创建 filename.p
```

MATLAB还提供对内存中存在的P码文件进行操作的命令，常见命令如下：

```
inmem                           列出内存中的所有 P 码文件
clear filename                  清除内存中的 filename.p 文件
clear functions                 清除内存中的所有 P 码文件
```

P码文件实例

例7.32 在MATLAB中，查看内存中的所有P码文件；然后清除所有P码文件，再次查看内存中的P码文件信息。

在MATLAB的命令窗口中输入下面的代码：

```
>> inmem
ans =
    'imformats'
    'workspacefunc'
    'num2str'
    'int2str'
>> clear functions
>> inmem
ans =
    'imformats'
```

如果在MATLAB的运行过程中，曾经打开或者编译某个M文件，则在内存中会保存其对应的P码文件。

变量传递

在编写程序的时候，参数传递一直是十分重要的问题。如何合理安排程序的变量传递直接关系到程序的效率，有时甚至关系到是否能够完成程序功能的问题。在MATLAB中，提供多种函数来实现变量检测、传递，同时也提供“变长度”输入输出变量。灵活使用这些命令可以完成多种复杂的功能，下面详细介绍。

7.8.1 变量检测命令

在MATLAB中，提供了[illegible]，用来检测输入和输出变量的个数，其语法格式如下：

- n = nargin　在函数体内，用于获取实际的输入变量
- [illegible]
- n = nargout　在函数体内，用于获取实际的输出变量
- [illegible]
- [illegible]
- [illegible]

例7.33　在MATLAB中，使用nargin函数检查函数的输入变量个数

Step 1　[illegible]，打开M文件编辑器，在M文件编辑器中输入如下的程序代码：

```
function foo(x,y)
if nargin > 2
    [illegible]
elseif nargin<2
    [illegible]
else
    %foo函数的主体，这里从略
end
```

Step 2　[illegible]，在MATLAB的命令窗口中输入"foo(1)"，[illegible]如图7.31所示。

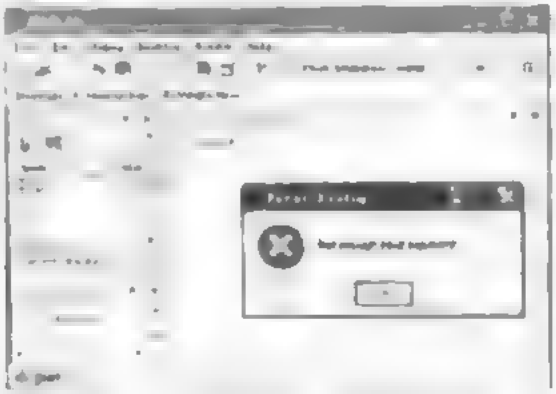


图7.31　输入参数太少的提示信息

Step 3　在MATLAB的命令窗口中输入"foo(pi,3,4)"，[illegible]：

```
>> foo(pi,3,4)
??? Error using ==> foo
Too many input arguments.
```

在MATLAB中，nargin、nargout为内置函数，不是变量，因此不能直接赋值，[illegible]

"变长度"变量函数

如果比较熟悉MATLAB的plot命令，就会发现一个问题：plot命令的输入变量个数是可变的，可以调用plot(x,y)作为默认格式，也可以使用最完整调用格式：

```
plot(x,y,'PropertyName1',PropertyValue1, 'PropertyName2',PropertyValue2...)
```

这个调用格式允许使用任意多的"属性名/属性值"对，设置绘制图形的属性。因此，plot函数为用户提供可以选择的可变长度变量。除了plot函数之外，MATLAB中还有很多函数可以接收"任意多输入"，返回"任意多输出"。同时，MATLAB也为用户自行编写"变长度"变量函数提供函数：

- varargin "变长度"输入变量列表，
- varargout "变长度"输出变量列表。

"变长度"变量实例

例7.34 在MATLAB中通过"变长度"变量，绘制不同半径、不同圆形的心形图。

step 1 单击MATLAB命令窗口工具栏中的□按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
function [ xcar,ycar] =drawcardioid(varargin)
%判断变长度变量的个数
error(nargchk(0,3,nargin));
nin = nargin;
%如果用户没有确定参数的数值
if nin==0
% 中心点的坐标数值
    cx1=0; cy1=0;
% 圆形的半径
    r=1;
%如果输入参数的数值个数为1
elseif nin==1
cx1=0; cy1=0;
r=varargin{1};
%如果输入参数的个数为2
elseif nin==2
cx1=0;
cy1= varargin{1};
r = varargin{2};
%如果函数输入参数的个数为3
else
cx1=varargin{1};
cy1=varargin{2};
r=varargin{3};
end

%判断输出参数的个数
if nargout==0
flag=1;
else
flag=0;
end
```

```
% 定义图形的角度变量数值
[illegible]
% 计算图形数据点的坐标数值
x0 = r*cos(theta);
y0 = r*sin(theta);
x1 = x0+cx1;
y1 = y0+cy1;
cx2 = 2*x0+cx1;
cy2 = 2*y0+cy1;
x3=x0.*cos(theta)-y0.*sin(theta);
[illegible]
[illegible]
x2(k,:) = cx2(k)+x0;
y2(k,:) = cy2(k)+y0;
xx(k)=cx2(k)+x3(k);
[illegible]
end
xcar = xx;
ycar = yy;
%绘制图形
[illegible]
plot(cx1,cy1,'mp','Markersize',6);hold on;
[illegible]
axis([cx1-3.2*r cx1+3.2*r cy1-3.2*r cy1+3.2*r]);
[illegible]
hold on
daspect([1 1 1]);
[illegible]
for k = 1:120
[illegible]
grid on;
[illegible]
end
end
```

step 2 [illegible]

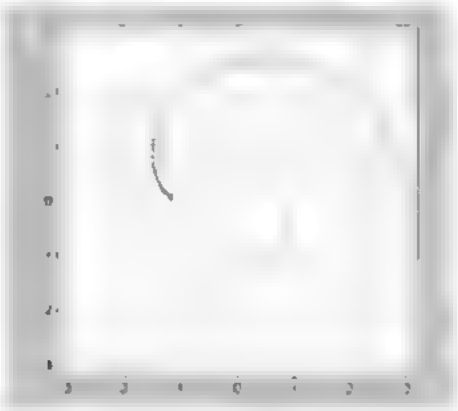


图 7.32　绘制中的动态图形

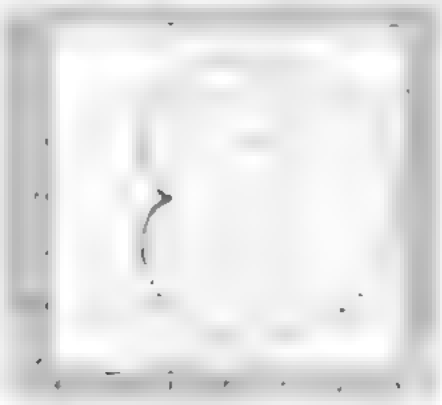


图 7.33 绘制完成的图形

说明

step 3

如图 7.34 所示。

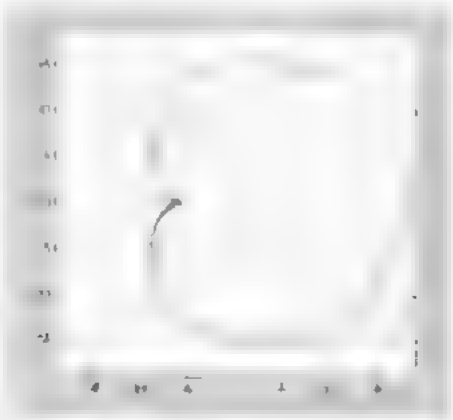


图 7.34 两个参量的绘制图形

说明

step 4

数值为 8。因此，上面程序代的绘制的是外层的小形图。

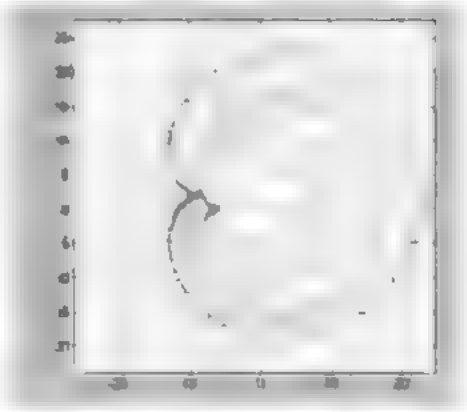


图7.35 绘制新的心形图

step 5 将数据导入，便于进一步数据处理。在MATLAB系统命令窗口中输入一系列代码

```
>> [x,y]=draw_heart(2,2,5);
>> coordinate=[x',y'];
```

当输入上面的代码后，输入"coordinate"，按"Enter"键，得到的结果如下

```
>> coordinate
coordinate =
   23.0000    2.0000
   22.9331    3.6876
   22.7331    5.3635
   22.4016    7.0160
   21.9413    8.6337
   21.3562   10.2055
   [illegible]   11.[illegible]
   19.8316   13.1694
   18.9048   14.5419
   17.8781   15.8294
   16.7600   17.0240
   15.5594   18.1186
   14.2861   19.1069
   12.9502   19.9838
   11.5623   20.7450
   ..........................          //由于篇幅，省略了部分数据结果
   [illegible]
   [illegible]
   [illegible]
   21.9413    4.6337
   [illegible]
   [illegible]
   [illegible]
   23.0000    2.0000
```

系统将来自draw_heart的数据赋予函数draw_heart的输出参量，在本例步骤中，没有选择显示输出参量，因此程序并没有得出输出参量。

在编写M文件时，函数声明行中的"变长度"变量，不仅有一般普通的变量之后，为了帮助读者"解密"变长度变量的工作原理，下面简要介绍varargin函数的工作原理。

在MATLAB中，varargin函数本身就是一个函数，当MATLAB调用包含varargin函数的M文件

varargin函数输入变量的分配规则是：输入变量按照先后顺序逐个对应分配给函数定义的输入变量与表中定义的固定变量，剩余的输入变量分配到varargin这个形数组中。因此，varargin函数的参数个数取决于分配到的输入变量数。

编写M函数文件时，varargin和varargout函数是实现可变参数"单调"输入变量处理varargin.m的工作原理，请对照varargin.m和varargout.m代码，查看关于varargin.m和varargout.m的参数输入变量和输出变量之间的关系。

7.8.4 跨空间计算表达式的数值

前面三种方法分别介绍了实现不同工作空间变量传递的两种方法——函数的输入输出变量和全局变量。在MATLAB中，还提供其他三种方法来实现跨空间的变量传递。本节介绍跨空间计算表达式数值的命令，详细的调用格式如下

- evalin(ws,expression)　跨空间计算出表达式的数值
- evalin(ws,expression1, expression2)　跨空间计算替代出表达式的数值。

例7.35　在MATLAB中，使用eval命令和evalin命令实现跨空间传递变量数值。

step 1　单击MATLAB命令窗口工具栏中的按钮，打开M文件编辑器，在M文件编辑器中输入下面的程序代码

```
function y=exeval(theta,arg)
t=(0:(length(theta)-1)/theta*2*pi;
y=subexeval(t,arg);

子函数代码
function yt=subexeval(theta,arg)
t=(0:(length(theta)-1)/theta*2*pi;argt='theta*exp(i*t)';
switch arg
case {'base','caller'}
   yt=evalin(arg,argt);
case 'self'
   yt=eval(argt);
end
```

step 2　将上面代码保存在"exeval.m"文件，再次单击MATLAB命令窗口工具栏中的按钮，打开新的M文件编辑器，在M文件编辑器中输入下面的程序代码

```
clear;
theta=5;
t=(0:(length(theta)-1)/theta*2*pi;
arg={'base','caller','self'};
for i=1:3
y=exeval(t,arg{i});
subplot(1,3,i);
plot(real(y),imag(y),'m','LineWidth',2)
axis square image;grid on
end
```

[illegible]

Step 3 [illegible]

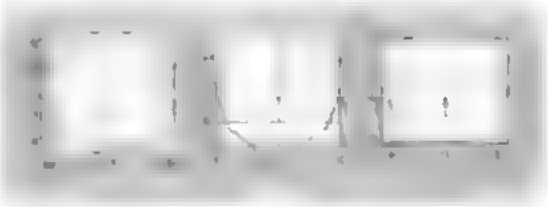


图 7.36 程序完成得到的图形结果

[illegible]

[illegible]

[illegible]到调用格式的机理。

当调用 eval(ws,expression)命令格式时，其执行机理如下：

- [illegible]
- [illegible]最数值。主调函数是相对于被调函数而言的。

[illegible]

7.8.5 跨空间赋值

[illegible]是跨空间赋值命令。其具体调用格式如下：

assignin(ws,'var',val)　跨空间向变量 var 赋值。

[illegible]

例 7.38　在 MATLAB 中使用 assignin 命令跨空间赋值。

Step 1 [illegible]的程序代码：

```
[illegible]
%显示信息对话框
prompt = {'Enter image name:','Enter colormap name:'};
```

```
% 定义标题
title = 'Image display - assignin example';
lines = 1;
% 定义变量的数值
def = {'my_image','hsv'};
% 获取答案
answer = inputdlg(prompt,title,lines,def);
assignin('base','imfile',answer{1});
assignin('base','cmap',answer{2});
```

step 2 将上面的代码保存为“assignin.m”文件，在MATLAB的命令窗口中，输入“assignin”，然后按“Enter”键，得到的结果如图7.37所示。

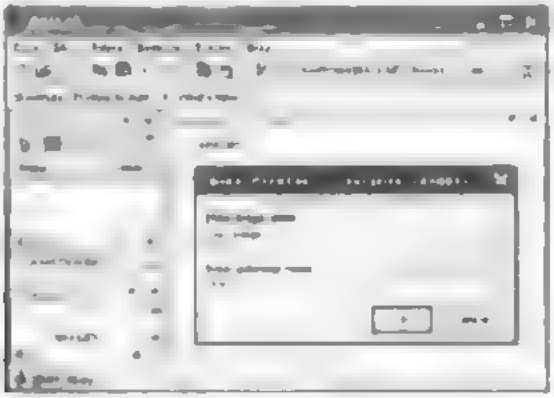


图7.37　显示程序代码的结果

step 3 单击对话框中的“OK”按钮，在MATLAB的命令窗口中，输入“whos”命令，查看程序中的所有变量，如图7.38所示。

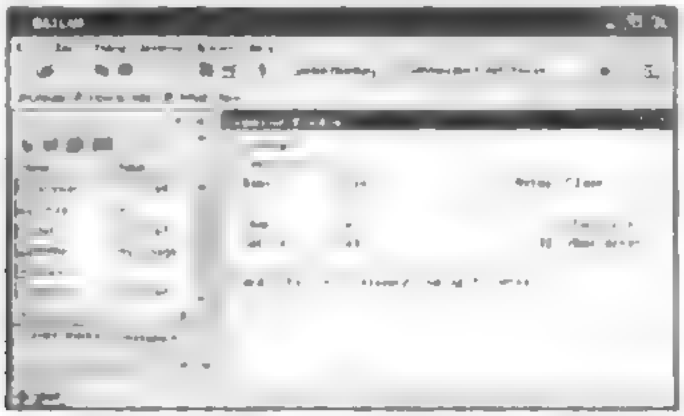


图7.38　查看程序的变量数值

在上面的程序代码中，相关的许多变量都是在程序代码中自行定义的，只有变量imfile和cmap是通过assignin命令赋值得到的。

7.9 字符串演算函数

在MATLAB中，[illegible] M文件，[illegible]。MATLAB [illegible] 函数。[illegible] 运算。

inline 函数

在 MATLAB 中，和字符串运算相关的函数主要有 feval、eval 和 inline 函数等。关于 eval 和 feval 函数的具体使用方法，请查阅"数值计算"章节的相关内容。在本小节中，将详细介绍内联函数 inline 的使用方法。

在 MATLAB 中，内联函数 inline 的调用格式如下：

- g = inline(expr) 将字符串表达式转换为输入变量自动生成的内联函数；
- g = inline(expr,arg1,arg2,...) 将字符串表达式转换为 arg1、arg2 输入变量自动生成的内联函数；
- g = inline(expr,n) 将字符串表达式转换为 x、p1、p2...pn 输入变量自动生成的内联函数。

MATLAB 还提供和 inline 函数相关的处理函数，主要有下面的函数：

- vectorize(inlilne_fun) 使内联函数适合数组运算的法则，
- char(inline_fun) 给出内联函数计算公式。

使用 inline 函数求解零点

例 7.37 在 MATLAB 中，求解超越函数 $f(t)=\sin^2 t\cdot e^{-at}-b|t|$ 的所有零点。

step 1 单击 MATLAB 命令窗口工具栏中的 □ 按钮，打开 M 文件编辑器。在 M 文件编辑器中输入下面的程序代码：

```
% 创建 inline 函数
y=inline('sin(t)^2*exp(-a*t)-b*abs(t)','t','a','b');
% 定义所有的变量
a=0.2;b=0.6;t=-10:0.01:10;
% 向量化 inline 函数
y_char=vectorize(y);
% 计算函数的数值
Y=feval(y_char,t,a,b);
% 绘制函数图形
clf,plot(t,Y,'r','Linewidth',2);hold on,plot(t,zeros(size(t)),'k');
xlabel('t');ylabel('y(t)'),grid,hold off
```

step 2 将上面的代码保存为"inline_zero.m"文件，返回 MATLAB 命令窗口，输入"ainline_zero"，然后按"Enter"键，得到的结果如图 7.39 所示。

step 3 获得函数的零点初始近似数值。上面的步骤显示函数零点的大致分布，现在需要获得近似初始数值。在 MATLAB 命令窗口中输入下面的命令：

```
[r,t]=ginput(5);
```

MATLAB 进入交互界面，可以动态地选取零点的近似数值，如图 7.40 所示。

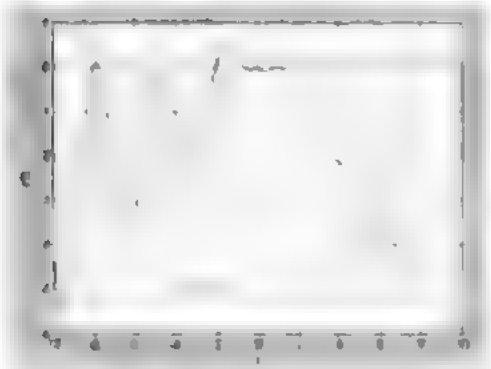


图 7.39 查看函数的零点分布

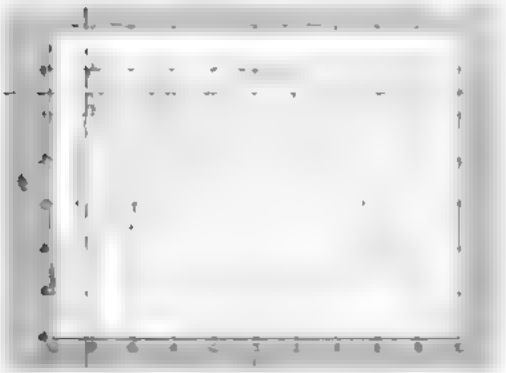


图 7.40 动态选取零点近似数值

step 4 查看选取的结果。当点取图形中的一个数据点后，MATLAB自动结束一段的程序代码，在MATLAB主命令窗口中查看数据点的坐标，得到如下结果如下：

```
>> r
r =
   -8.3180
   -7.5346
   -2.2350
   -0.8065
    0.3917
```

step 5 得到函数的精确零点数值。在MATLAB命令窗口中输入下面的程序代码：

```
>> for ii=1:5
[s(ii),m(ii),exitflag]=fzero(y,r(ii),[],4,5);
end
```

step 6 查看计算结果。在上面的程序代码中，s表示函数精确的零点，m是对应的函数数值，用户可以在MATLAB中查看其相应的数值如下：

```
>> s
s =
   -8.0386   -7.7420   -2.0222   -0.6010   -0.6010
>> m
m =
  1.0e-015 *
   -0.8882         0    0.2220         0         0
```

[illegible]

793 使用 inline 函数绘制图形

例 7.38 在 MATLAB 中，绘制陀螺运动的图形。

Step 1 [illegible] 的程序代码

```
function tlyd;
% tlyd陀螺运动的微分方程
[illegible]
    '-a*y(4)*y(6)*sin(y(1))+b*g*sin(y(1));',...
    [illegible]
    '*y(6);y(2)*y(4)*[ sin(y(1))-(a-2)*cot(y(1))*cos(y(1))] ',...
    [illegible]
% plot(t,Y)
axis([-1 1 -1 1 0 1]);
[illegible]
% 获得陀螺侧面的三维数据网格(由参数坐标 x0,y0,z0 为参数)
[x0,y0,z0]=cylinder(0:.05:.5,60);
[illegible]
% 以下四句是陀螺底面参数网格(由参数坐标 cx0,cy0,cz0 为参数)
Q=linspace(0,2*pi,60);
cx0=0.5*cos(Q);
cy0=0.5*sin(Q);
[illegible]
% 陀螺初始位置侧面数据(静坐标 x,y,z 随陀螺的运动而变化)
[x,y,z]=zbbh(x0,y0,z0,th1,ph1,ps1);
[illegible]
% 画初始位置陀螺侧面并获取的画手形的句柄
[illegible]
colormap(winter);
% 画初始位置陀螺之底面并获取的画手形的句柄
h2=fill3(cx,cy,cz,[1 0.62 0.40]);
% 在绘图执行之前,暂停0.5秒
[illegible]
for i=1: length(u);
    % 求陀螺新的侧面数据
    [illegible]
    % 求陀螺新位置底面数据
```

```
    [cx,cy,cz]=zbbh(cx0,cy0,cz0,u(i,1),u(i,3),u(i,5));
    %更新位置陀螺轨迹
    [illegible]
    %更新位置陀螺底面
    [illegible]
    drawnow
end;
[illegible]
%----------坐标变换的子函数----------
function [x,y,z]=zbbh(x0,y0,z0,thi,phi,psi)
x=[illegible]
    +y0*(-sin(psi)*cos(phi)-cos(psi)*cos(thi)*sin(phi))...
    [illegible]
y=x0*(cos(psi)*sin(phi)+sin(psi)*cos(thi)*cos(phi))...
    [illegible]
z=[illegible]
```

[illegible]坐标转换的函数，[illegible]中，[illegible]函数，而其中zbbh则是被调函数，也称是子函数。

step 2 在MATLAB[illegible]，输入命令[illegible]，[illegible]键，[illegible]图7.41所示。

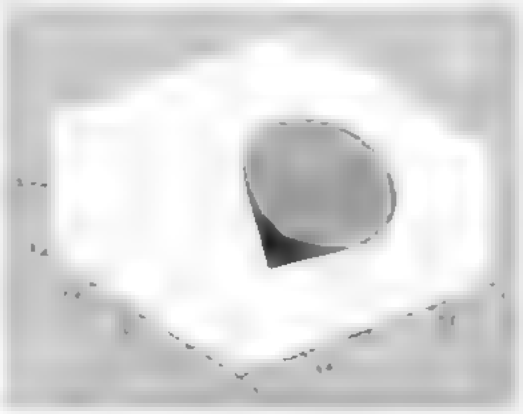


图7.41 陀螺运动的开始图形

[illegible]MATLAB的中[illegible]，[illegible]的结束图形，如图7.42所示。

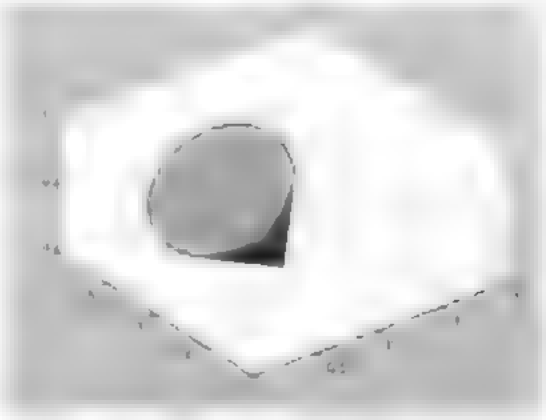


图7.42 陀螺运动的结束图形

在上面的程序代码中，首先使用 inline 函数定义陀螺运动的微分方程，这就避免了在后面的程序代码中使用字符串带来不必要的麻烦。

实例 使用 inline 函数求解极值

例 7.39 在 MATLAB 中，使用内联函数求解函数 $f(x,y)=100(y-x^2)^2+(1-x)^2$ 在定义域范围内的极小值。

step 1 在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> ff=inline('100*(x(1)-x(2)^2)^2+(1-x(1))^2','x');
>> x0=[-1.2,1];
```

step 2 查看使用单纯形求解的极小值点。在命令窗口中输入下面的代码：

```
>> [sx,sf,sexit,soutput]=fminsearch(ff,x0)
sx =
    1.0000    1.0000
sf =
  2.0520e-009
sexit =
     1
soutput =
    iterations: 86
     funcCount: 163
     algorithm: 'Nelder-Mead simplex direct search'
       message: [1x196 char]
```

step 1 查看使用拟牛顿法求解的极小值点。在命令窗口中输入下面的代码：

```
>> [ux,sf,uexit,uoutput,grid,hess]=fminunc(ff,x0)
Warning: Gradient must be provided for trust-region method;
  using line-search method instead.
> In fminunc at 241
Optimization terminated: relative infinity-norm of gradient less than
options.TolFun.
Computing finite-difference Hessian using user-supplied objective
function.
ux =
    1.0000   -1.0000
sf =
  2.3431e-012
uexit =
     1
uoutput =
        iterations: 21
         funcCount: 84
          stepsize: 1
    firstorderopt: 2.1546e-005
        algorithm: 'medium-scale: Quasi-Newton line search'
          message: 'Optimization terminated: relative infinity-norm of
gradient less than options.TolFun.'
grid =
  1.0e-004 *
```

```
    0.[illegible]
   -0.2155
hess =
  202.0000  400.0241
  400.0241  800.2919
```

Step 4 为了方便查看函数的图形，[illegible]一个函数的曲面图形。在MATLAB中[illegible]输入下面的代码

```
>> x=[illegible];
>> [X,Y]=meshgrid(x,y);
>>Z=100*(Y-X.^2).^2+(1-X).^2;
>> [illegible];
>> shading interp
[illegible]
```

Step 5 查看绘图结果，输入上面的命令后，按“Enter”键，程序[illegible]

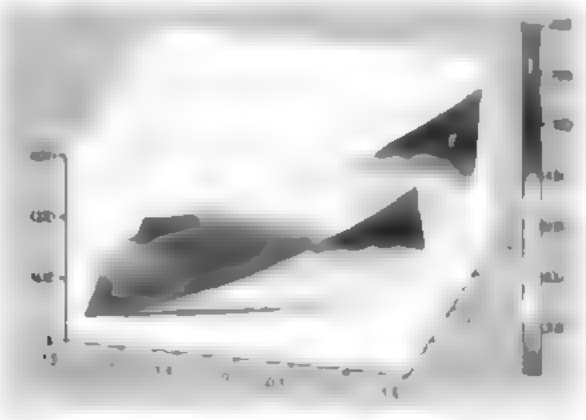


图 7.43 函数的图形

从上面的图形中可以看出，该函数有一个极小值，[illegible]函数的图形修改，大多算法难以接近这个极小值。这个函数被称作著名的Rosenbrock“banana”测试函数

7.10 程序的调试和剖析

和其他编程语言一样，一个由MATLAB编写的工作程序中，通常[illegible]的难度，主要在于输入动模或者多个参数[illegible]，掌握和调试[illegible]相结合对编写工作程序[illegible]重要。

一般来讲，程序设计的错误主要有三类：语法错误、[illegible]逻辑错误。[illegible]两种。其中，语法错误通常在程序编译阶段[illegible]，MATLAB会在[illegible]编辑器中给出[illegible]错误并改正。

对于逻辑错误，情况[illegible]，处理起来也[illegible]。逻辑错误一般[illegible]步骤：[illegible]程序设计是[illegible]，[illegible]理解等。逻辑错误[illegible]

较多，如程序运行正常，但是结果异常，或者程序代码不能正常运行而中断。逻辑错误相对于语法错误而言，更难查找错误原因，因为逻辑错误一般是动态的，一旦停止运行，中间变量都会被删除，很难跟踪程序变量的变化情况。

针对上面两种错误类型，本节将详细介绍直接调试法和工具调试法两种常见的调试（Debug）方法。同时，除了介绍如何进行程序调试之外，还介绍如何对程序代码进行解析，以便对程序运行的时间开销进行分析，以改善程序性能。

直接调试法

根据前面章节的介绍，读者已经发现MATLAB语言本身的特点，该语言的向量化程度比较高，程序设计一般都显得比较简单。MATLAB本身的运算能力强，命令系统比较简单，可读性比较强，因此直接调试法往往十分有效。通常采用的直接调试法包括如下手段：

- 经过分析，将重点怀疑语句行或者命令行后面的分号去掉，或者改成逗号，使得运算结果显示在屏幕上面；
- 在有疑问的语句附近，添加显示某些关键变量值的程序语句，查看变量数值；
- 在程序的适当位置添加keyboard命令，将MATLAB执行到相应的程序代码时，会暂停执行程序代码，同时在命令窗口中显示k>>提示符，用户可以查看或者修改变量的数值；在提示符后面输入return命令行后，系统会返回到程序代码中，继续执行原文件。

直接调试法实例

例7.40 在MATLAB中使用直接调试法，调试程序代码。

step 1 单击MATLAB命令窗口工具栏中的□按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
function f=ballw(K,ki)
% ballw.m 演示红色小球沿一条封闭螺线运动的实时动画
% 仅演示实时动画的调用格式为 ballw(K)
% 既演示实时动画又拍摄照片的调用格式为 f=ballw(K,ki)
% K 红球运动的循环数（不小于 1 ）
% ki 指定拍摄照片的瞬间，取 1 到 1034 间的任意整数
% f 存储拍摄的照片数据，可用 image(f.cdata) 观察照片
t1=(0:1000)/1000*10*pi;x1=cos(t1);y1=sin(t1);z1=-t1;
t2=(0:10)/10;x2=x1(end)*(1-t2);y2=y1(end)*(1-t2);z2=z1(end)*ones(size
(x2));
t3=t2;z3=(1-t3)*z1(end);x3=zeros(size(z3));y3=x3;
t4=t2;x4=t4;y4=zeros(size(x4));z4=y4;
x=[x1 x2 x3 x4];y=[y1 y2 y3 y4];z=[z1 z2 z3 z4];
plot3(x,y,z, 'y','Linewidth',2 ), axis off % 绘制曲线
% 定义 " 线 " 色、" 点 " 型（点）、点的大小（ 40 ）、擦除方式（ xor）
h=line( 'Color' ,[0.67 0 1], 'Marker' , '.' , 'MarkerSize' ,40, 'EraseMode'
, 'xor' );
% 使小球运动
n=length(x);i=1;j=1;
while 1 % 无穷循环
set(h, 'xdata' ,x(i), 'ydata' ,y(i), 'zdata' ,z(i));
drawnow; % 刷新屏幕 <21>
pause(0.0005) % 控制球速 <22>
```

```
i=i+1;
[illegible]
if (i==ki&j==1);f=getframe(gcf); end % 拍摄 i=ki 时的照片 <25>
end
if i>n
i=1;j=j+1;
if j>K; break ; end
end
end
```

step 2 [illegible] 如图 7.44 所示。

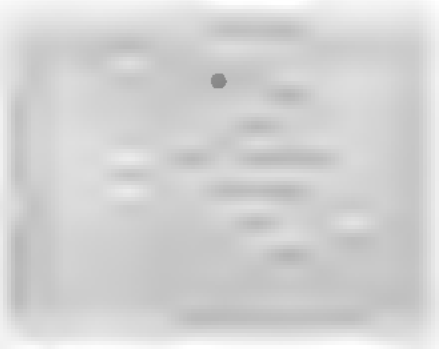


图 7.44 程序运行后的结果

当 MATLAB 完成上面的程序代码后，得到的结果如图 7.45 所示。

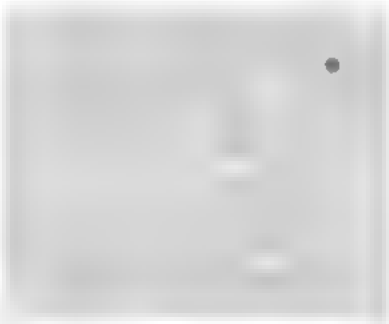


图 7.45 程序结果的图形

[illegible]

step 3 [illegible]

```
[illegible]
[illegible]
[illegible]
[illegible]
x=[x1 x2 x3 x4];y=[y1 y2 y3 y4];z=[z1 z2 z3 z4];
data=[x',y',z']                    // 添加显示封闭曲线的坐标数值
[illegible]
if i>n
i=1; [illegible]
```

```
if j>K; break ; end
end
end
```

修改上面的程序代码后，将程序代码保存为 "ballw.m" 文件。

step 4 查看程序结果。返回命令窗口，输入命令行 "ballw(2,200)"，得到的结果如下：

```
>> ballw(2,200)
data =
    1.0000         0         0
    0.9995    0.0314   -0.0314
    0.9980    0.0628   -0.0628
    0.9956    0.0941   -0.0942
    0.9921    0.1253   -0.1257
    0.9877    0.1564   -0.1571
    0.9823    0.1874   -0.1885
    0.9759    0.2181   -0.2199
    0.9686    0.2487   -0.2513
    0.9603    0.2790   -0.2827
   ........................................    //限于篇幅，省略了部分数据
         0         0  -15.7080
         0         0  -12.5664
         0         0   -9.4248
         0         0   -6.2832
         0         0   -3.1416
         0         0         0
         0         0         0
    0.1000         0         0
    0.2000         0         0
    0.3000         0         0
    0.4000         0         0
    0.5000         0         0
    0.6000         0         0
    0.7000         0         0
    0.8000         0         0
    0.9000         0         0
    1.0000         0         0
```

从上面的程序结果中可以看出，当在程序代码中添加一个简单的语句 "data=[x',y',z']" 后，就可以在程序代码执行的过程中，查看封闭曲线的所有坐标值数值。如果程序结果中封闭曲线不正常，则可以从上面的数据中查看数值的问题。

step 5 显示小球位置的坐标数值。打开上面步骤保存的 "ballw.m" 文件，将程序代码修改如下：

```
function f=ballw(K,ki)
.........................................//保持该部分代码不变
while 1 % 无穷循环
set(h, 'xdata' ,x(i), 'ydata' ,y(i), 'zdata' ,z(i));
bw=[ x(i),y(i),z(i)]                          //计算小球位置的坐标数值
.........................................// 保持该部分代码不变
if i>n
i=1;j=j+1;
if j>K; break ; end
end
end
```

修改上面的程序代码后，将程序代码保存为“ballw.m”文件。

step 6 查看程序结果。返回到命令窗口，输入命令行“ballw(2,200)”，得到的结果如下：

```
bw =
    0.2000   -0.0000  -31.4159
bw =
    0.1000   -0.0000  -31.4159
bw =
         0         0  -31.4159
bw =
         0         0  -31.4159
bw =
         0         0  -28.2743
..............................................// 限于篇幅，省略了部分数据
bw =
    0.6000         0         0
bw =
    0.7000         0         0
bw =
    0.8000         0         0
bw =
    0.9000         0         0
bw =
     1     0     0
```

在上面的步骤中，分别使用简单程序代码查看关键的程序数据，如果在程序运算过程中出现问题，则可以从上面的程序数值中查找相应的问题。

工具调试法

上面演示了如何使用直接调试法来调试程序代码，但是如果函数文件规模较大，文件的内嵌复杂，或者有较多函数、子函数、私有函数待调用，直接调用法则可能会失败，这个时候则需要使用MATLAB的专门调试工具——调试器。

MATLAB自带的M文件编辑器同时也是程序代码的编辑器，可以在M文件编辑器中输入程序代码后，直接在其中进行调试，显得方便和直观。

关于M文件编辑器的功能在前面介绍过，在本小节中将需要介绍该编辑器的调试功能键和菜单选项。为了显示M文件编辑器的调试功能，有必要调用某个M文件。在本小节中，调用的是7.1.2小节中编写的“test_sort.m”文件，调用的结果如图7.46所示。

在上面的文件调试器中，当设置M文件进入调试阶段时，M文件编辑器提供的调试功能键都会被激活，表示可以使用这些功能键来调试程序代码。为了介绍方便，M文件的调试功能键图形如图7.47所示。

将上面的调试功能键从左到右依次命令为1、2…7，其对应的功能介绍如下：

- **1号功能键**：功能是断点设置（或者清除），对应的菜单选项为“Debug” ⇨ “Set/clear breakpoints”，快捷键为F12，对应的MATLAB命令为dbstop/dbclear。
- **2号功能键**：功能是清除全部断点，对应的菜单选项为“Debug” ⇨ “Clear Breakpoints in All Files”，对应的MATLAB命令为dbclear all。

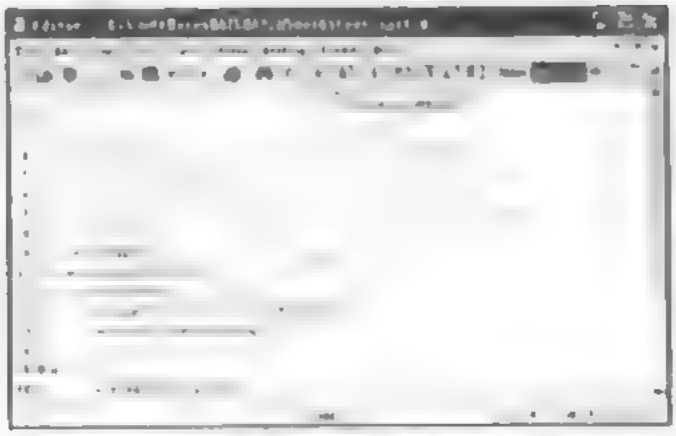


图 7.46　M 文件调试器

图 7.47　调试功能键

- **3 号功能键**：[illegible]，[illegible]"Debug"→"Step"，快捷键为 F10，对应的 MATLAB 命令为 dbstep。
- **4 号功能键**：[illegible]，对应菜单项为"Debug"→"Step In"，快捷键为 F11，对应的 MATLAB 命令为 dbstep in。
- **5 号功能键**：[illegible]，[illegible]"Debug"→"Step Out"，快捷键"Shift+F11"，对应的 MATLAB 命令为 dbstep out。
- **6 号功能键**：[illegible]，对应菜单项为"Debug"→"Continue"，快捷键为 F5，对应的 MATLAB 命令为 dbcont。
- **7 号功能键**：[illegible]"Exit Debug Mode"，对应的 MATLAB 命令为 dbquit。

以上[illegible]。除了[illegible]而外，MATLAB 还提供其他[illegible]，[illegible]"Debug"→"Set Breakpoint"，MATLAB 会弹出"MATLAB Editor"对话框，如图 7.48 所示。

图 7.48　设置条件断点

[illegible]"Warnings...."，打开对应的对话框，如图 7.49 所示。

在 MATLAB 文件[illegible]外，还有一个"工具栏"[illegible]，如图 7.50 所示。

图 7.49 设置程序代码的警告信息

图 7.50 工作空间切换

7.10.4 工具调试法实例

下面通过一个实例来说明如何使用 MATLAB 的工具调试代码程序。

例 7.41 [illegible] 在 MATLAB 调试代码程序。

Step 1 [illegible] 果如图 7.51 所示。

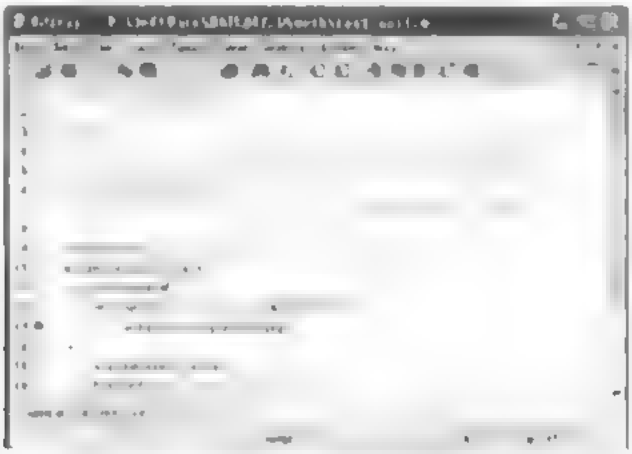


图 7.51 添加程序的断点

Step 2 [illegible] 程序代码的调试工作。得到的结果如下

```
Enter number of the numbers to sort:2
14        array(ii)=input(string);
K>> return
Enter value 1:2
[illegible]
K>> return
Enter value 2:3
 sorted data:
  2.0000
```

```
3.0000
```

[illegible]

以修改进行。

Step 5 [illegible]

按钮，查看程序代码的运行情况如下

```
Enter number of the numbers to sort:3
13          array(ii)=input(string);
Enter value 1:2
14  end
[illegible]
13          array(ii)=input(string);
Enter value 2:1
14  end
K>> return
13          array(ii)=input(string);
Enter value 3:5
14  end
K>> return
 sorted data:
  1.0000
  2.0000
[illegible]
```

[illegible]

7.10.5 程序剖析

[illegible]

- [illegible]
- profile off　暂停程序剖析器的运行。
- [illegible]
- profile clear　清除以往的剖析记录
- profile viewer　开启界面式的程序剖析器。

[illegible]

程序剖析实例

例 7.42 选用自行编写的M文件sierpinskihm.m，对其进行程序代码的剖析。

step 1 打开已经编写完成的"sierpinskihm.m"，其代码显示如下：

```
function sierpinskihm(n);
% Sierpinskinhm海绵
% 调用格式:sierpinskihm(n);
% n为迭代次数
x=0; % x为初始正方形的第一顶点的横坐标
y=0; % y为初始正方形的第一顶点的纵坐标
z=0; % z为初始正方形的第一顶点的竖坐标
d=1; % d为初始正方形的边长
x4=x;
y4=y;
z4=z;
d4=d;
x2=[];
y2=[];
z2=[];
a=[0,1,2];
[x1,y1]=meshgrid(a);
[x1,z1]=meshgrid(x1,a);
[y1,z1]=meshgrid(y1,a);
x1=[x1(1,1:4),x1(1,6:9),x1(2,[1,3,7,9]),x1(3,1:4),x1(1,6:9)];
y1=[y1(1,1:4),y1(1,6:9),y1(2,[1,3,7,9]),y1(3,1:4),y1(1,6:9)];
z1=[z1(1,1:4),z1(1,6:9),z1(2,[1,3,7,9]),z1(3,1:4),z1(3,6:9)];
for q=1:n;
   for p=1:length(x);
       x3=x(p)+d/3*x1;
       y3=y(p)+d/3*y1;
       z3=z(p)+d/3*z1;
       x2=[x2,x3];
       y2=[y2,y3];
       z2=[z2,z3];
   end
   d=d/3;
   x=x2;
   y=y2;
   z=z2;
end
axis([x4,x4+d4,y4,y4+d4,z4,z4+d4])
P1=[d,d,d,d,d];
P2=[0,d,d,0,0];
P3=[0,0,d,d,0];
P4=[0,0,0,0,0];
for p=1:length(x);
   patch(x(p)+P1,y(p)+P2,z(p)+P3,z(p)+P3);
   patch(x(p)+P2,y(p)+P4,z(p)+P3,y(p)+P4);
   patch(x(p)+P2,y(p)+P1,z(p)+P3,x(p)+P2);
   patch(x(p)+P2,y(p)+P3,z(p)+P4,y(p)+P3);
   patch(x(p)+P2,y(p)+P3,z(p)+P1,x(p)+P4);
   patch(x(p)+P4,y(p)+P2,z(p)+P3,z(p)+P3);
end
```

step 2

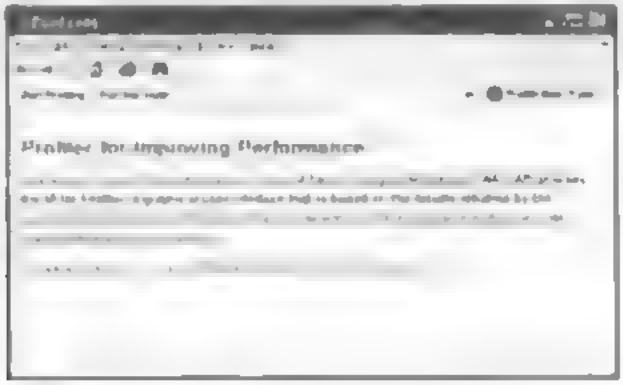


图 7.52 程序代码剖析器

step 3

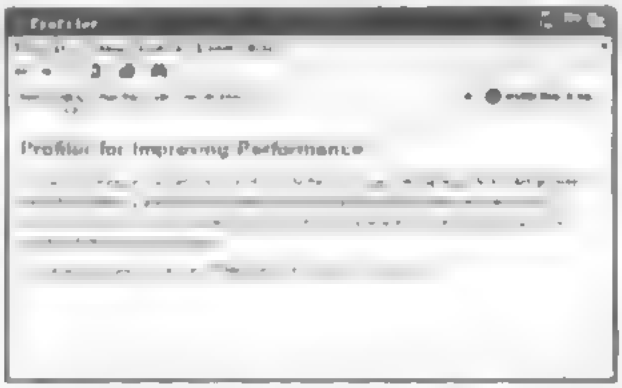


图 7.53 运行程序代码的剖析

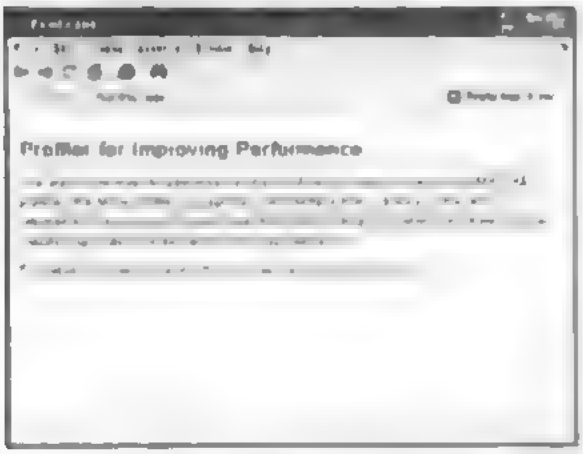


图 7.54 剖析程序当中

图 7.55 剖析结果

[illegible]，即在上面的结果[illegible]了该程序代码的"分析汇总表 | Profile Summary"[illegible]中查看关于该程序的代码剖析结果。当程序运行结束后，[illegible]，如图 7.56 所示。

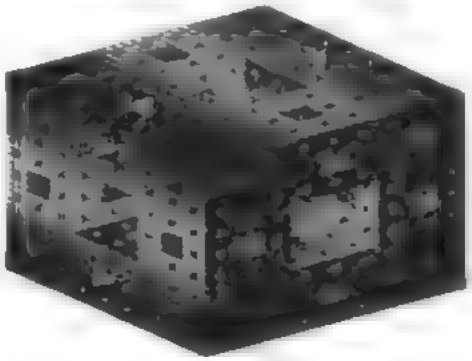


图 7.56 程序运行的结果

step 4 分析代码剖析结果。上面的[illegible]，[illegible]要希望了解比较详细或者细节的内容，可以单击界面中带有[illegible]"sierpinskihm"字样，查看对应的详细内容。其中关于剖析的内容如图[illegible]。

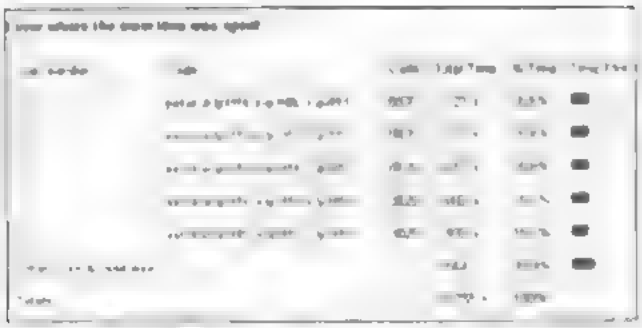


图 7.57　程序代码中最消耗时间的程序行

[illegible]，MATLAB 会[illegible]，如图 7.58 所示。

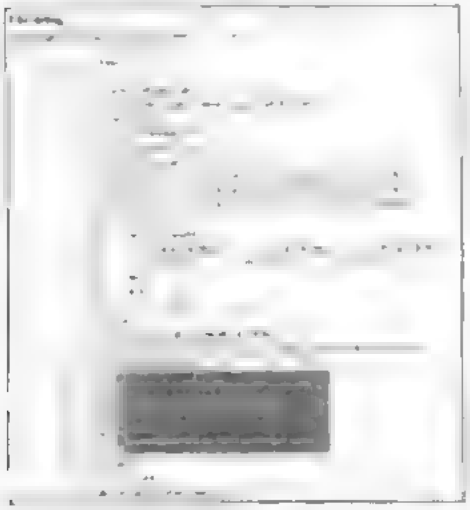


图 7.58　程序代码的性能分析列表

[illegible]"瓶颈"位置，[illegible]关键程序行。

7.11　小结

本章主要介绍了在 MATLAB 环境中进行程序设计的基础知识，主要包括 MATLAB 变量、表达式、MATLAB 程序结构、[illegible]，这些内容是在 MATLAB 环境中进行科学计算和数据处理的基础知识。在后面章节中，将介绍 Simulink 仿真系统。

◆　　◆　　◆

第8章 Simulink 仿真系统

本章包括

- Simulink 的基础知识
- Simulink 的基础操作
- Simulink 的属性设置
- 非线性系统建模
- Simulink 的数据类型
- Simulink 的信号
- 线性系统建模

在 MATLAB 中，Simulink 是用来建模、仿真和分析动态多维系统的交互工具。可以使用 Simulink 提供的标准模型库或者自行创建模型库，描述、模拟、评价和精化系统行为。同时，Simulink和MATLAB之间的联系十分便捷，可以使用一个灵活的操作系统和应用广泛的分析和设计工具。最后，除了可以使用 Simulink 建模和仿真之外，还可以通过其他软件包产品来完成更多的分析任务。

在本章中，首先使用一个例子来说明 Simulink 的仿真创建过程，然后介绍 Simulink 的工作环境和常见工具。由于 Simulink 的内容比较繁多，本章将主要介绍关于 Simulink 的基础知识，包括线性系统、非线性系统和离散系统的建模方法。最后，本章还将介绍关于 Simulink 分析工具的相关知识，这样，就可以更方便地使用该工具分析 Simulink 创建的模型对象。关于 Simulink 的更深层次的应用知识，将在后面的章节中详细介绍。

8.1 Simulink 的基础知识

Simulink 是一个复杂的应用系统。为了让读者更直观地了解 Simulink 的使用方法和操作界面方法，在本节将首先介绍关于 Simulink 的一些基础知识，包括 Simulink 概述和安装知识，创建方法等。

8.1.1 Simulink 概述

和 MATLAB 的其他组件（有时也会被称为软件包）相比，Simulink 的一个突出特点就是它完全支持图形用户界面（GUI），这样就极大地方便了用户的操作方法。用户只需要进行简单的拖曳操作就可以构造出复杂的仿真模型，它的外观以方块图形的形式来呈现，而且采用分层结构。从建模的角度来看，这种方法可以让用户将主要的精力放在具有创造性的算法和模块结构的设计上，而不用将精力放在算法的实现上。从分析研究的角度来看，Simulink 模型可以让用户知道具体环节的动态细节，而且还可以让用户清晰地了解到各系统组件、各子系统、各系统之间的信息交换。

在 Simulink 环境中，用户可以观察到现实世界中摩擦、风阻等非线性或者随机因素对系统行为的影响，同时可以在仿真过程中改变需要观察的参数数值，观察系统行为的变化。这样，可以摆脱复杂的数学推演和烦琐的程序代码，直接探索各种因素的影响。

在 MATLAB7.0 中，可以直接在 Simulink 环境中运行的工具包很多，包括通信、控制、信号、电力等各个领域，所涉及的内容也比较广泛和专业。如果用户合理地使用这些工具包中的内容，可以创建各种复杂的仿真模型，实现各种复杂的功能。在本章后面的部分内容中，将会涉及这些工具包的内容，在对应的地方将详细介绍其内容。

[illegible]

[illegible]

8.1.2 安装 Simulink

[illegible]

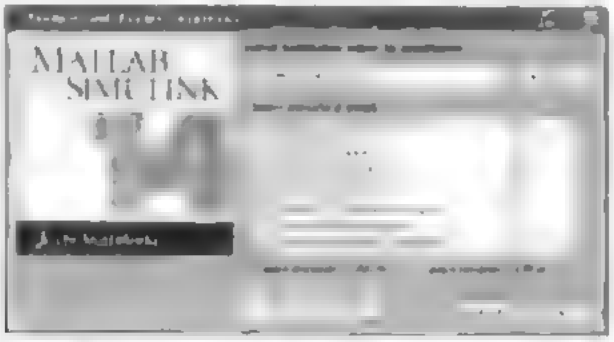


图 8.1 选择安装 Simulink 的相关组件

[illegible]

- Signal Processing Toolbox
- SimMechanics
- SimPowerSystems
- [illegible]
- Simulink Control Design
- Stateflow
- Real-Time Workshop
- Virtual Reality Toolbox

[illegible]

[illegible]

图 8.2 显示隐藏算组件

8.1.3 启动 Simulink

[illegible] MATLAB 命令窗口中的 Simulink 按钮，如图 8.3 所示。

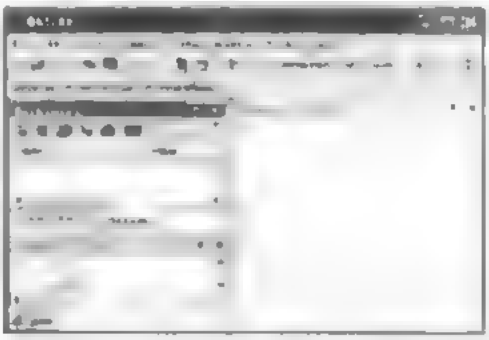


图 8.3 使用按钮启动 Simulink

除了上面的方法外，还可以使用下面两种方法：

- 在 MATLAB 的命令窗口中输入命令 ">> Simulink"。
- 选择 MATLAB 编辑栏中的 "File" → "New" → "Model" 命令。

[illegible] 选择查看各种 Simulink 模块，如图 8.4 所示。

图 8.4 Simulink 模块库浏览器

[illegible]

[illegible]

[illegible]

8.1.4 添加 Simulink 模块

[illegible] Simulink 的创建过程和步骤。

例 8.1 [illegible]

Step 1 单击"Simulink Library Browser"对话框中的按钮，或者点击菜单"File"→"New"→"Model"命令，打开一个空白模型窗口，如图 8.5 所示。

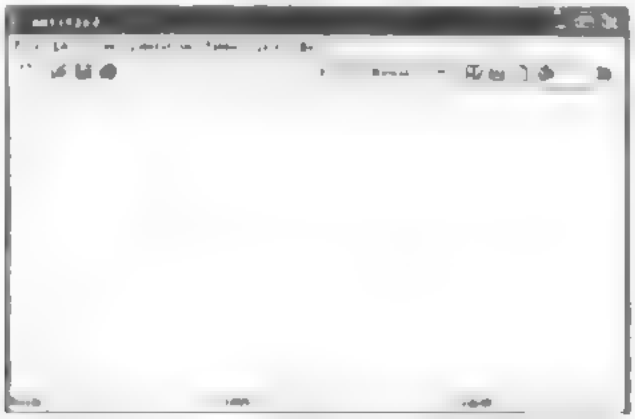


图 8.5 新建模型窗口

Step 2 选择"Signal"信号源，选择"Simulink Library Browser"对话框中的"Sources"模块组，然后在右侧窗口中选择"Chirp Singal"模块，如图 8.6 所示。

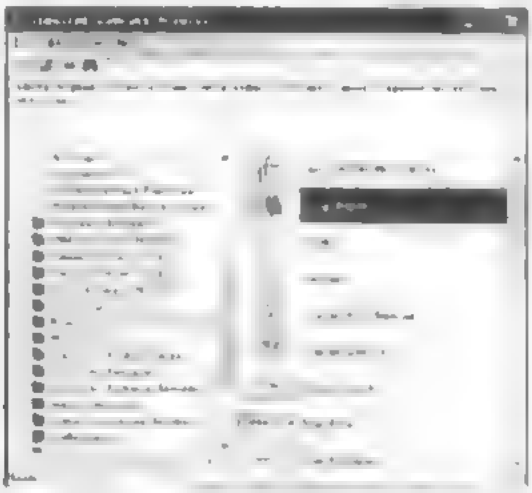


图 8.6 选择"Chirp Singal"信号源

当在"Simulink Library Browser"模块浏览器窗口中选择相应的信号源后，在对话框的上半部分会显示关于该信号的简单介绍，通过这些基础的介绍大致了解该信号的内容。

Step 3 在"Chirp Singal"信号源上，选择"Chirp Singal"模块，按下鼠标左键，将其拖至新建模型窗口中，如图 8.7 所示。

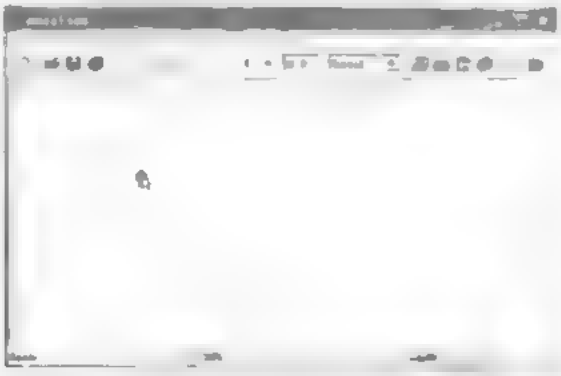


图 8.7 添加"Chirp Singal"信号

Step 4 查看添加的"Chirp Singal"信号源，选择添加的信号源，按下鼠标左键，在鼠标光标位置处会显示用户添加的信号模块，如图 8.8 所示。

图 8.8 添加信号模块

当创建模型窗口中添加模块后，对该模块的名称由"untitled"修改为"untitled"，而对于添加的模块则会按照默认设置来入手，用户可以根据需要修改模块的属性。

8.1.5 设置模块的属性

延续上面小节的步骤，

Step 1 编辑"Chirp Singal"模块的属性，选择"Chirp Singal"模块，将模块拖放到模块上，按下鼠标并进行拖动，将该模块放置到合适位置，选择菜单栏中的"Format"→"Foreground Color"→"Green"命令，将模块的背景颜色设置为绿色，完成效果如图。

Step 2 设置"Chirp Singal"模块的参数，在模块窗口的"Chirp Singal"模块，双击"Block Parameters:Chirp Singal"对话框，设置该模块对应的参数，如图 8.9 所示。

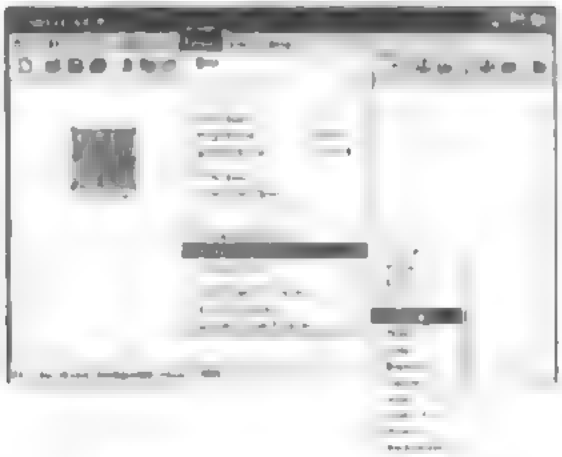


图8.9 设置模块的外观属性

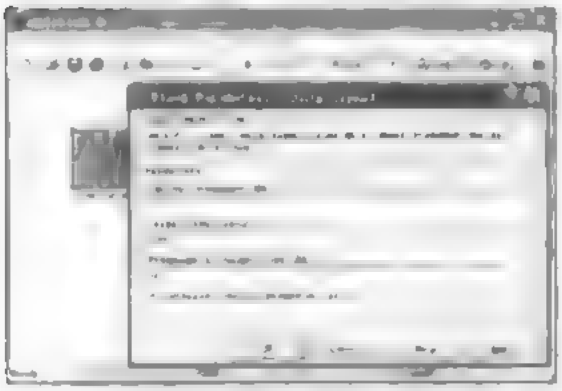


图8.10 设置模块参数

[illegible]

step 3 在“Sine Wave”模块上单击鼠标右键，[illegible]模块，设置模块的外观属性，得到的结果如图8.11所示。

图8.11 添加“Sine Wave”程序模块

step 4 设置一个 [illegible] 模块的属性，[illegible] 模块。[illegible] 函数的帮助文件，如图 8.12 所示。

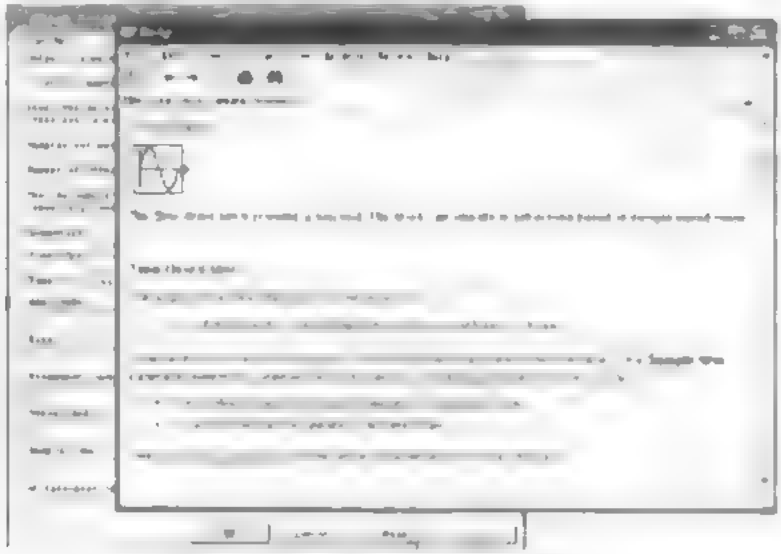


图 8.12 设置模块的属性

step 5 添加数学运算符模块，设置其[illegible]属性。[illegible] 模块。[illegible] 模块，并设置其外观属性，得到的结果如图 8.13 所示。

图 8.13 添加数学运算符

step 6 添加[illegible]模块，并设置其外观属性。[illegible] 模块。[illegible] 模块，并设置其外观属性，得到的结果如图 8.14 所示。

图8.14　添加显示屏模块

8.1.6 连接模块

按照上面小节的步骤。

Step 1 连接信号模块。将鼠标指向"Uniform Random Integer"模块的右侧输出端，当鼠标变成十字形时，按住鼠标左键，将其拖至"Add"模块的右侧第一数据输入端，如图8.15所示。

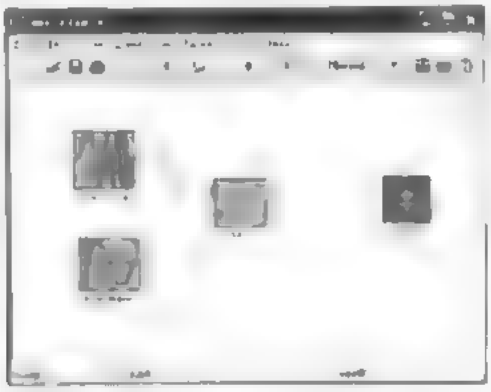


图8.15　连接程序模块

Step 2 连接其他程序模块。按照上面步骤中的方法，连接其他信号模块，并调整各模块的相对位置，得到的结果如图8.16所示。

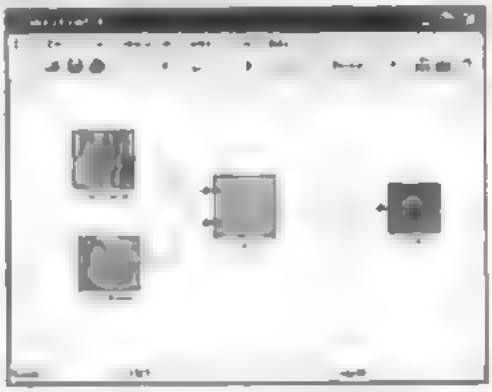


图8.16　完成连接的程序模块

[illegible]

8.1.7 运行仿真系统

继续上面小节的步骤

step 1 查看仿真结果。单击模型窗口中的"仿真启动"按钮，或者在[illegible]"Start"命令，进行模型的仿真，然后双击示波器模块中的[illegible]，查看[illegible]号波形，如图 8.17 所示。

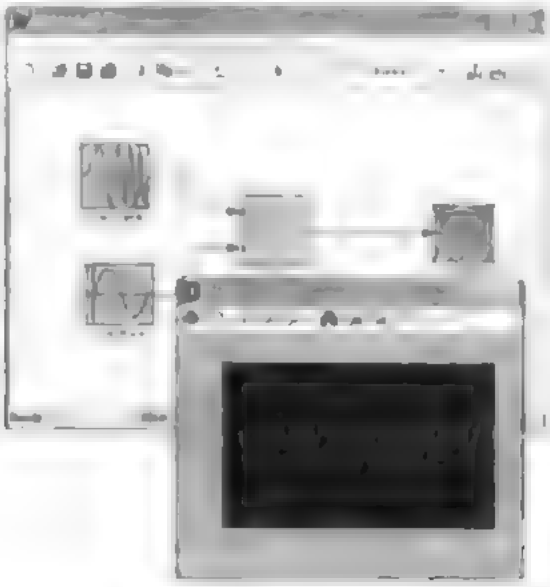


图 8.17 查看仿真结果

step 2 修改仿真显示的结果。单击[illegible]"Scope"菜单，将波形充满整个坐标框，如图 8.18 所示。

图 8.18 修改仿真显示的结果

step 3 修改仿真的参数。在默认情况下，模型[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，重新进行仿真，观察[illegible]

图 8.19　修改仿真参数

8.1.6 模块库浏览器

[illegible]

图 8.20　"Simulink Library Browser"对话框

[illegible]

图 8.21　对话框的工具栏

[illegible]

- ◆ **1 号按钮：**标准的 Windows 工具菜单，表示新建一个 Simulink 模型。
- ◆ **2 号按钮：**标准的 Windows 工具菜单，表示打开一个已经创建的 Simulink 模型。
- ◆ **3 号按钮：**[illegible]

◆ 4号按钮：[illegible]会在对应的跨度中进行查找。

[illegible]使用该项功能，如图8.22所示。

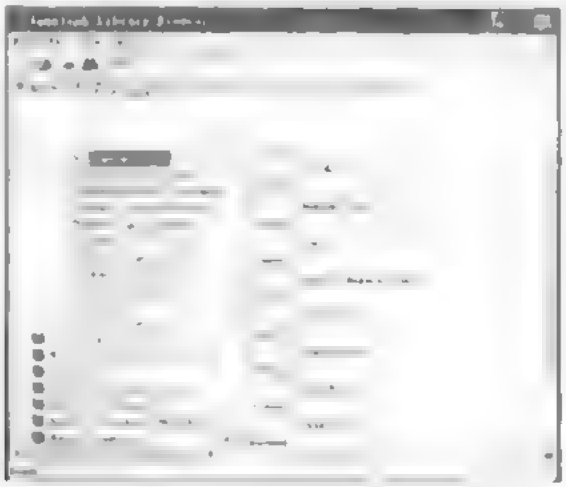


图8.22 模块搜索

[illegible]钮。依次循环查看查找的多个结果。

[illegible]

[illegible]

[illegible]

8.1.9 Simulink 模型窗口界面

[illegible]

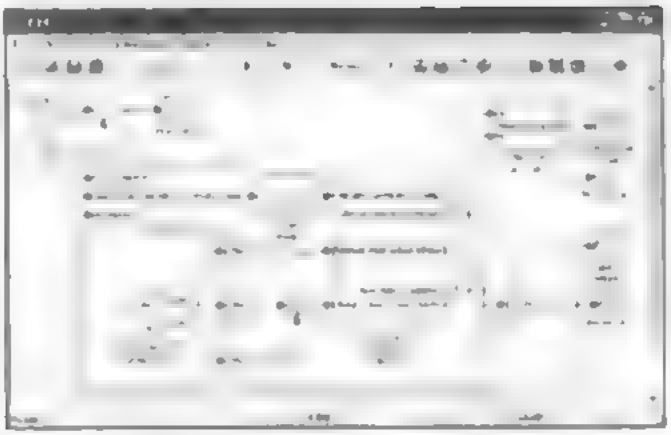


图 8.23　Simulink 模型窗口界面

[illegible]

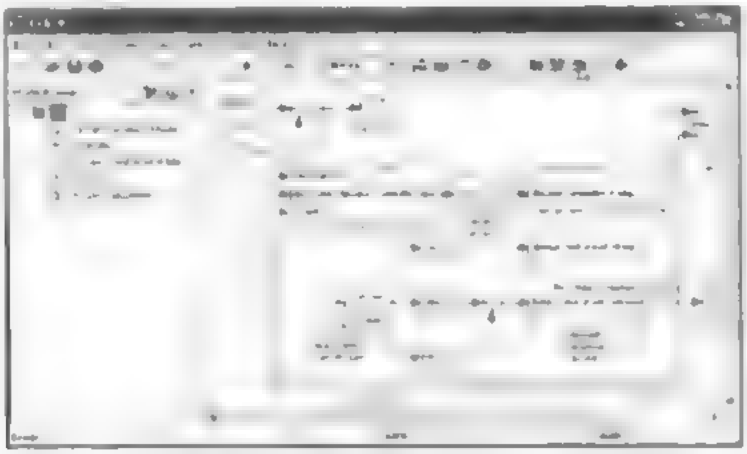


图 8.24　双窗口形式

根据[illegible]菜单栏、工具栏、编辑栏和状态栏，下面将详细介绍这些组成部分的功能。

其中，模型窗的工具栏如图 8.25 所示。

图 8.25　模型窗的工具栏

[illegible]

- **按钮 1～7**：[illegible]新建、打开、保存、打印、剪切、复制和粘贴模块。
- **按钮 8**：撤销上一个操作步骤。
- **按钮 9**：返回下一个操作步骤
- **按钮 10**：[illegible]
- **按钮 11**：[illegible]

- 文字框 1：[illegible]
- 下拉菜单 1：[illegible] 加速模式。用户可以根据需要选择对应的加速模式。
- 按钮 12：准备系统的仿真。
- 按钮 13：产生 RTW 程序代码。
- 按钮 14：刷新整个系统。
- 按钮 15：更新整个系统。
- 按钮 16：为子系统产生程序代码。
- 按钮 17：显示 Simulink 的模块库浏览器。
- 按钮 18：打开模块管理器。
- 按钮 19：打开或者隐藏模型浏览器。
- 按钮 20：转到上一级系统中。
- 按钮 21：打开调试器。

上面工具栏中的各[illegible]对象，MATLAB 各[illegible]具体信息。

在模型窗口的状态栏中，会显示整个仿真系统的状态，从左到右依次为[illegible]，状态栏右的文字的含义依次如下。

- Ready：表示模型[illegible]，可以开始仿真。
- 100%：表示编辑框的显示比例。
- ode45：表示仿真选用的积分算法是"ode45"。

除了上面的参数之外，[illegible]模型的[illegible]动态仿真信息。

在 MATLAB 的模型窗口，[illegible]菜单选项[illegible]，[illegible]Windows 标准菜单的功能。

8.1.10 "File"菜单

"File"菜单的主要选项以及对应的功能如表 8.1 所示。

表 8.1 "File"菜单选项和功能

主要的菜单选项	功能
Model Properties	设置 Simulink 的模型属性
Preferences	设置 Simulink 界面形态的默认属性
Source Control	设置 Simulink 和 SCS 的接口
Print	打印模型
Print Details	[illegible]

对于其他子菜单的功能，在本书的[illegible]，[illegible]模型[illegible]，[illegible]，选择菜单栏中的"File"→"Print Details"命令，[illegible]"Print Details"对话框。可以在该对话框中设置相关的选项参数，然后单击其中的"OK"按钮，如果

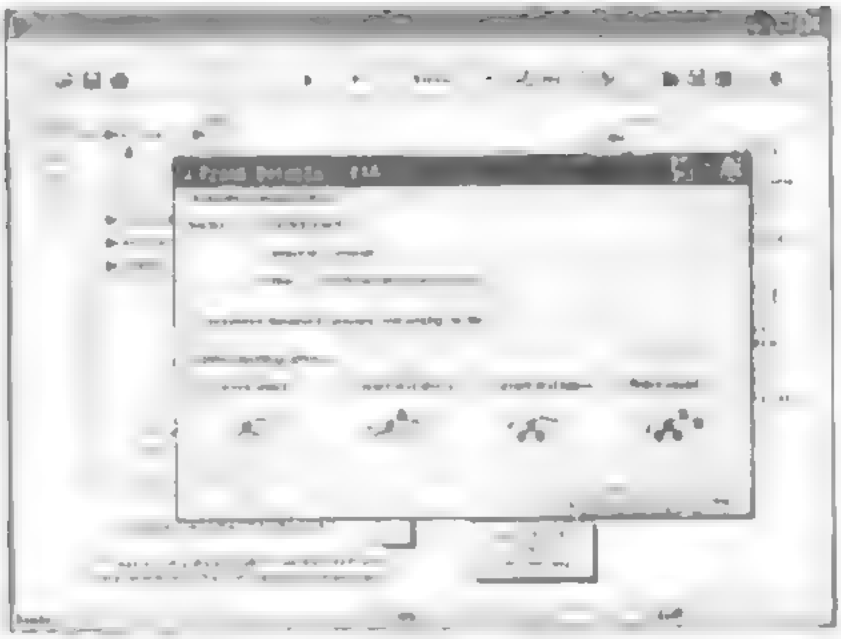


图 8.26 生成模型报告文件

8.1.11 "Edit" 菜单

"Edit" 菜单的主要选项以及对应的功能如表 8.2 所示。

表 8.2 "Edit" 菜单选项和功能

主要菜单选项	功能
Find	搜索 Simulink 系统内的模型块、信号、注解等等对象
Block Properties	打开模块属性对话框
Create Subsystem	创建子系统
Mask Subsystem	封装子系统
Look under Mask	查看被封装子系统的内部结构
Signal Properties	查看信号属性
Edit Mask	编辑封装子系统
Subsystem Parameters	设置子系统的参数
Mask Parameters	封装过的子系统的参数设置
Update Diagram	更新模型框图的外观属性

[illegible]

[illegible] 命令，打开 "Find - f14" 对话框，如图 8.27 所示。

图 8.27 搜索对话框

在上面的对话框中，可以在"Find in"下拉列表中选择搜索范围，然后在"Search for"文本框中输入搜索的关键词。如果搜索范围是多个系统或多个子系统，则需要在"Start in system"下拉菜单中选择搜索范围。最后，在"Match"选项卡中选择搜索匹配方式，单击对话框中的"Find"按钮，进行搜索，如图 8.28 所示。

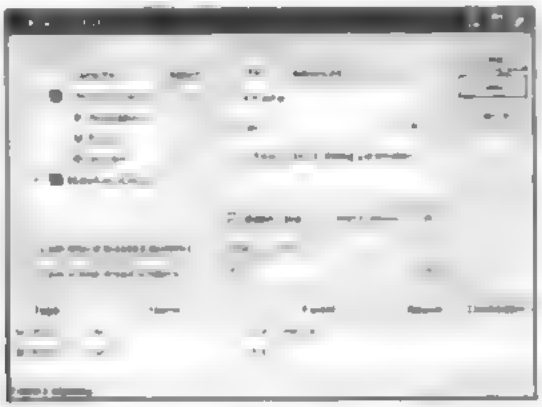


图 8.28 搜索结果

从上面的搜索结果中，既可单击"Highlight found"选项，也可选择"Match whole word"或"Match case"方式，或在"Look inside masked systems"选项，在搜索对话框"Search for"关键词为"sin"的搜索结果只有两个结果。

保持搜索关键词不变，修改其他搜索参数，可以查看新的搜索结果，如图 8.29 所示。

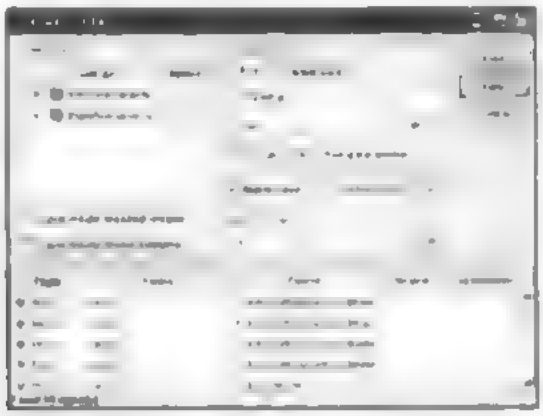


图 8.29 修改搜索条件

从上面的搜索结果可以看出，保持"Sin"关键词不变，修改搜索条件后，搜索结果的变化发生很大的变化，得到的搜索结果有 13 个。

8.1.12 “View”菜单

“View”菜单的主要选项以及对应的功能如表8.3所示。

表8.3 “View”菜单选项和功能

主要的菜单选项	功能
Block Data Tips Options	设定当鼠标移到某个模块时，是否显示提示信息
Library Browser	打开模块库浏览器
Port Values	设置[illegible]鼠标操作，显示模块端口的当前值
Model Explorer	打开模块资源管理器，进行模块资源管理

[illegible]，选择菜单栏中的“View”→“Model Explorer”命令，打开“Model Explorer”对话框，如图8.30所示。

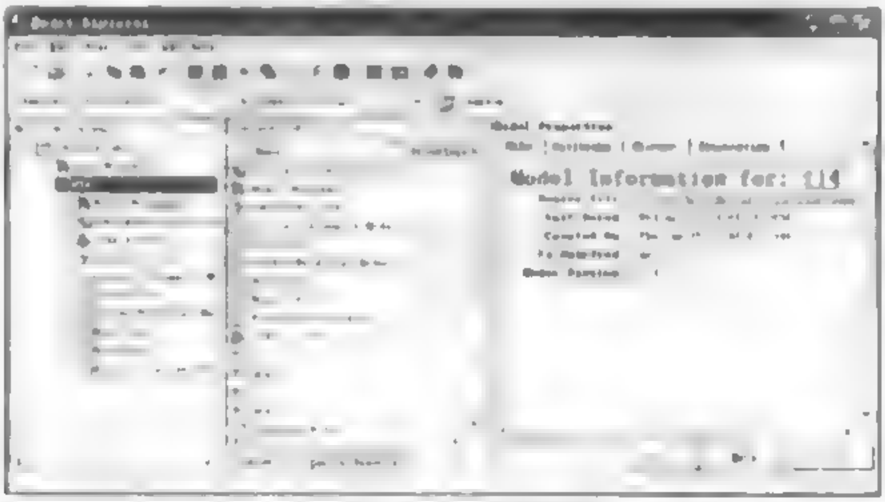


图8.30 模型资源管理器

[illegible]主要函数。[illegible]面板中查看对应对象的各种属性。

在“[illegible]”[illegible]选择“[illegible]”选项，查看对应的属性内容，如图8.31所示。

图8.31 查看对象的属性内容

[illegible]

[illegible]

图 8.32 显示"Customize Contents"菜单

[illegible]

[illegible]

8.1.13 "Simulation"菜单

[illegible]

在上面的"模拟数据"对话框中，主要有"Solver"、"Data Import/Export"、"Diagnostics"、"Hardware Implementation"和"Model Referencing"等选项卡，[illegible]数值。

[illegible]

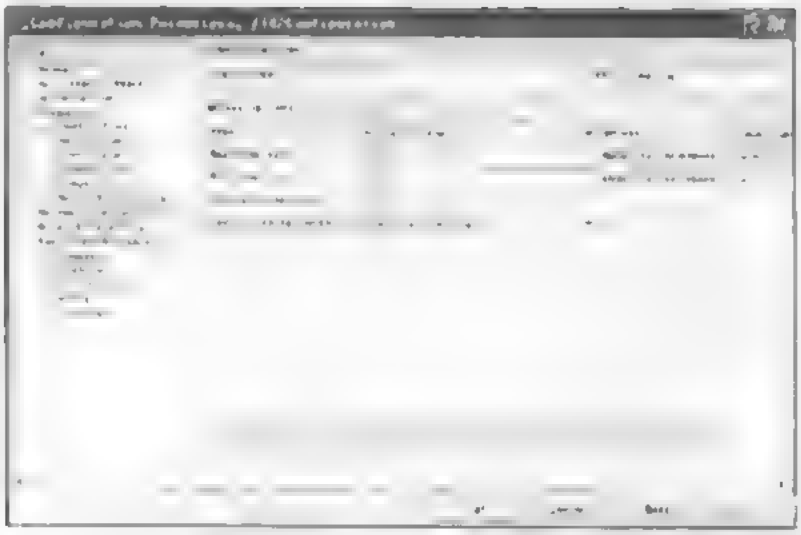


图 8.33　仿真参数对话框

8.1.14　"Help" 菜单

"Help" 菜单的主要选项以及对应的功能如表 8.4 所示。

表 8.4　"Help" 菜单选项和功能

主要的菜单选项	功能
Using Simulink	显示关于 Simulink 的帮助部分
Blocks	显示按字母排列的 Blocks 帮助部分
Block Support Table	显示模块所支持的数据类型的帮助内容
About Simulink	显示 Simulink 版本信息

[illegible]

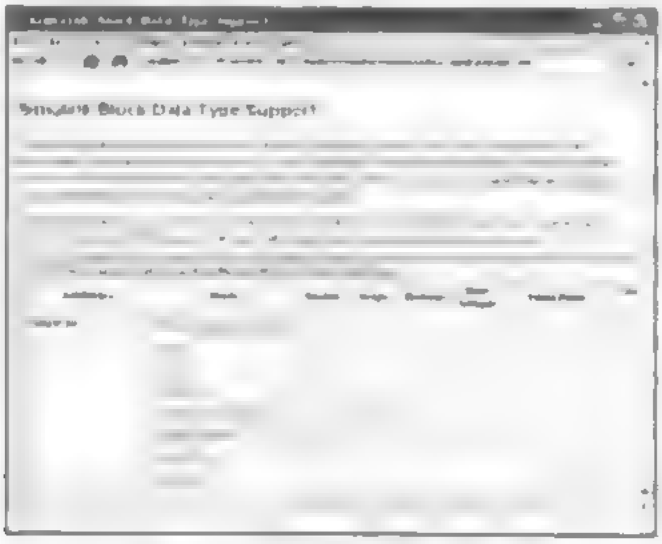


图 8.34　模型所支持的数据类型

[illegible]

8.2 Simulink 的数据类型

[illegible] Simulink 中信号和模块参数的数据类型。

[illegible] 模型的数据类型，一般都将是 Double 数据类型。

8.2.1 Simulink 支持的数据类型

[illegible] 是指 MATLAB 自行定义的数据类型。

[illegible]

表 8.5 MATLAB 的内置数据类型

数据类型	说明
[illegible]	[illegible]
Single	单精度浮点类型
Int8/Uint8	有符号/无符号的 8 位整数
Int16/Uint16	有符号/无符号的 16 位整数
Int32/Uint32	有符号/无符号的 32 位整数

[illegible] 如图 8.3 所示。

如果在打开参数设置对话框中没有特定的数据类型，表明该模块可为多种数据类型的输入。如 Double 数据类型。

[illegible] 如图 8.3 所示。

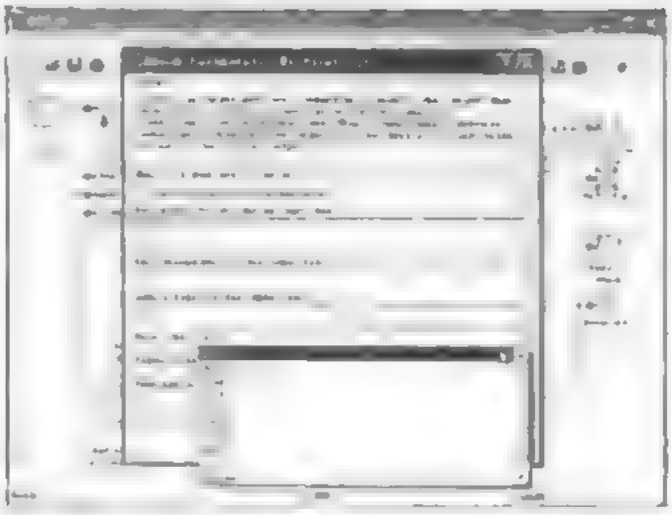


图8.35 查看模块所支持的数据类型

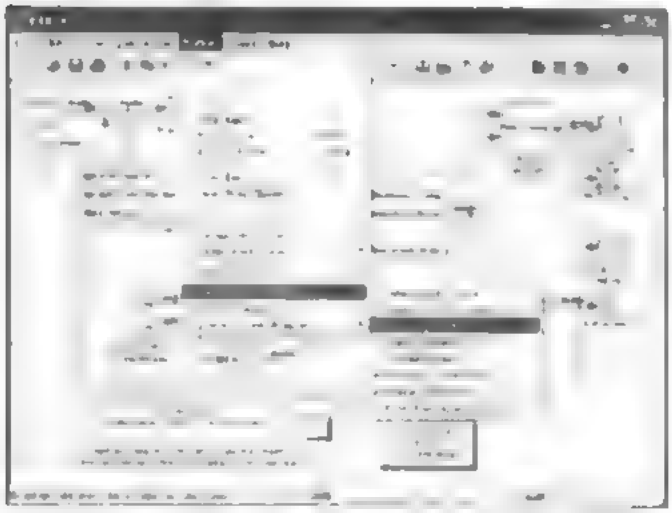


图8.36 查看各模块的数据类型

[illegible]

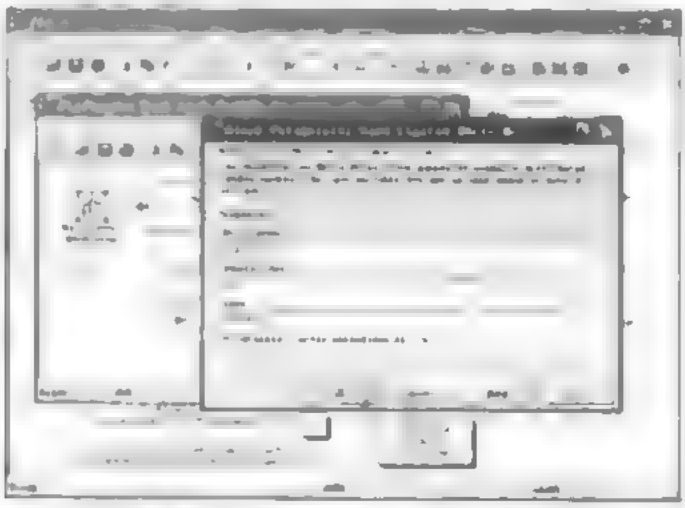


图8.37 设置模块参数的数值

在上面的对话框中，在对话框文本框中输入"[0 0]"，来模块的参数设置为[0 0]，以手动定义点数据类型。如果模块不支持用户指定的数据类型，则会MATLAB会弹出相应的警告信息。

8.2.2 Simulink 中的数据传递

[illegible]

8.2.3 Simulink 中的数据转换实例

例 8.2 使用简单的数学算子模块（>模块）的数据类型转换功能

step 1 打开 Simulink 的空白模型编辑器，从其中选取"Constant"数值模块，将其数值设置为"0"，同时打开其参数管理器，将其数据类型选择为"Boolean"，如图 8.38 所示。

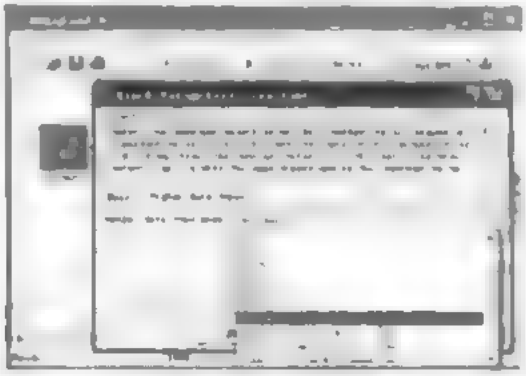


图 8.38 添加参数模块并设置属性

step 2 再加入一个常数数值模块，将其数值设置为"1"，将其数据类型设置为"Boolean"，得到的结果如图 8.39 所示。

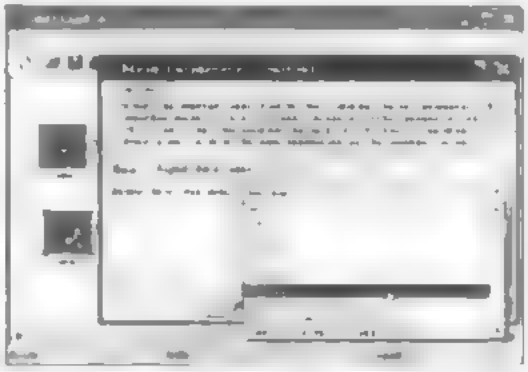


图 8.39 添加下一个参数模块

[illegible]

step 3 [illegible] 模块，然后添加“Scope”模块，如图 8.40 所示。

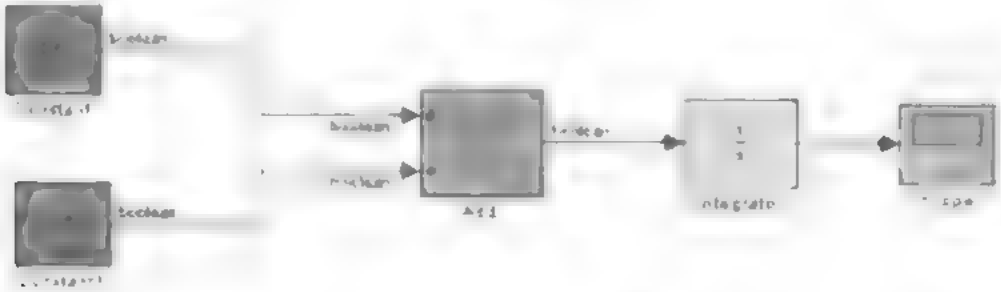


图 8.40 添加的模块系统

step 4 [illegible] 提示对话框如图 8.41 所示。

图 8.41 错误提示信息

[illegible]

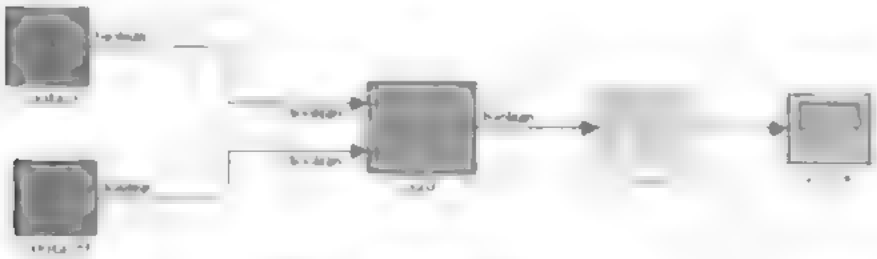


图 8.42 高亮显示错误模块

Step 5 [illegible]

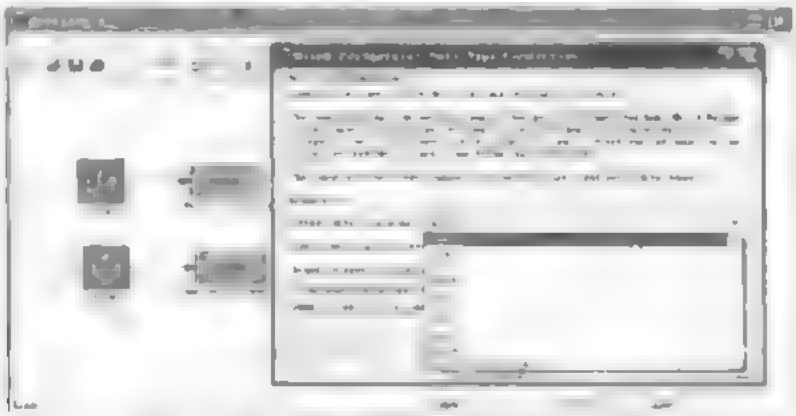


图 8.43 添加数据转换模块

[illegible]

在 Simulink 中，[illegible]

Step 6 [illegible]

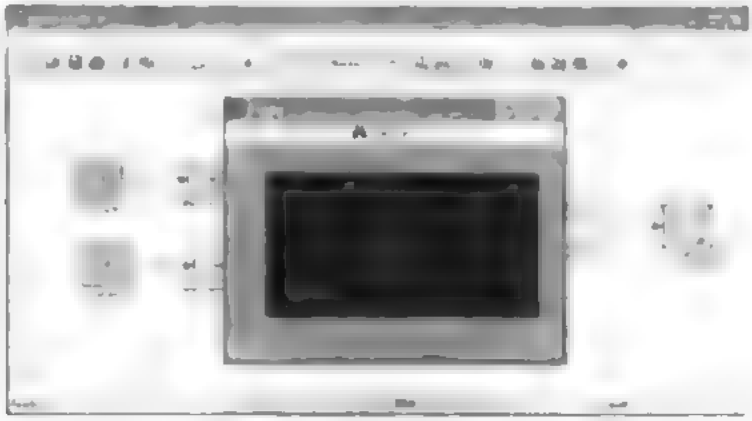


图 8.44 查看仿真结果

从上面的仿真结果图可以看出，[illegible]

8.2.4 向量化模块

[illegible]

例 8.3 使信号模块输出向量信号。

Step 1 [illegible] 的参数设置器，在其中设置信号的参数，如图 8.45 所示。

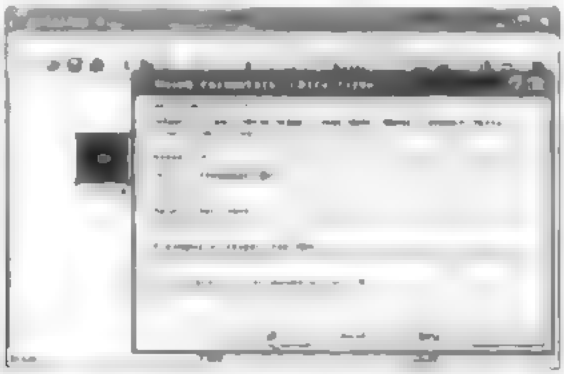


图 8.45 设置模块的属性

在上面的对话框中，在"Initial frequency"文本框中输入向量"[0.1 0.4]"，在"Target time"文本框中输入向量"[1 [illegible]]"，在"Frequency at target time"文本框中输入向量"[1 2]"，然后单击对话框中的"OK"按钮，完成向量的输入。

在 [illegible] 中，[illegible] 模块产生的是一个线性信号，[illegible] 模块的线性信号[illegible]

Step 2 [illegible] 模块上单击，"Scope"模块，运行仿真，再双击"Scope"模块，查看仿真结果，[illegible] 的结果如图 8.46 所示。

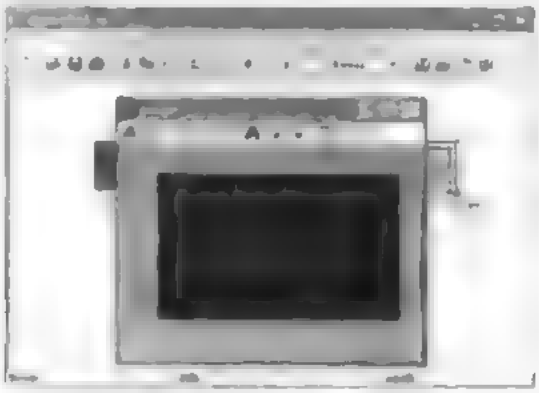


图 8.46 查看仿真结果

[illegible]

8.2.5 使用 Mux 模块

例 8.4 使用“Mux”模块输出向量信号。

step 1 打开新的空白模型窗口，在其中添加“Chirp Signal”模块，然后双击该模块，打开相应的参数设置窗，在其中设置信号的参数，如图 8.47 所示。

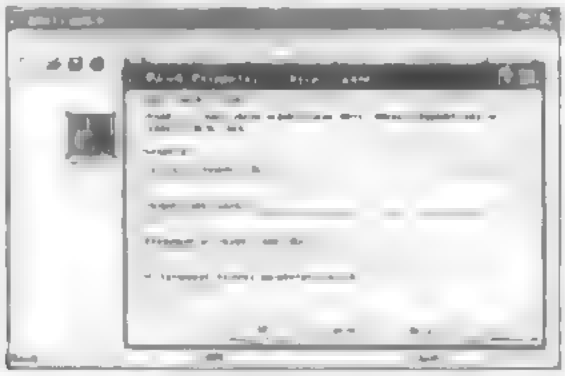


图 8.47 添加“Chirp Signal”信号

step 2 在其中添加“Sine Wave”信号，双击该信号模块，打开相应参数设置窗，在其中设置信号的参数，如图 8.48 所示。

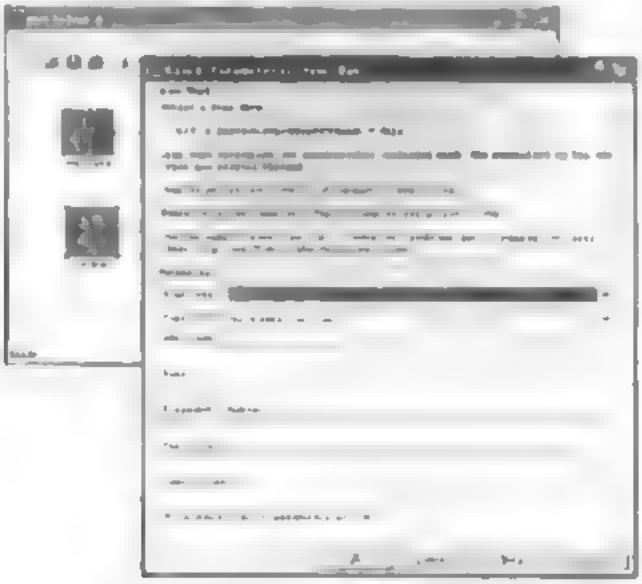


图 8.48 添加“Sine Wave”信号

step 3 添加“Mux”模块，将[illegible]模块的[illegible]模块，[illegible]“Type”模块，显示向量化处理的结果，如图 8.49 所示。

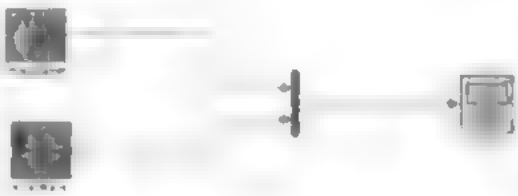


图 8.49 添加“Mux”模块

[illegible]

step 4 设置“Mux”模块的属性。[illegible]

图 8.50 设置“Mux”模块的属性

step 5 查看仿真结果。[illegible]，查看仿真的结果，得到的结果如图 8.51 所示。

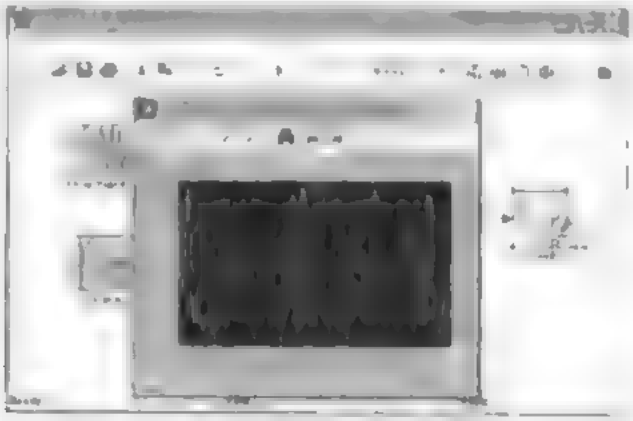


图 8.51 查看仿真结果

[illegible]

[illegible]

8.2.6 标量扩展

[illegible]

例 8.5 通过“乘/除”模块来显示输入扩展功能。

step 1 [illegible]

[illegible]

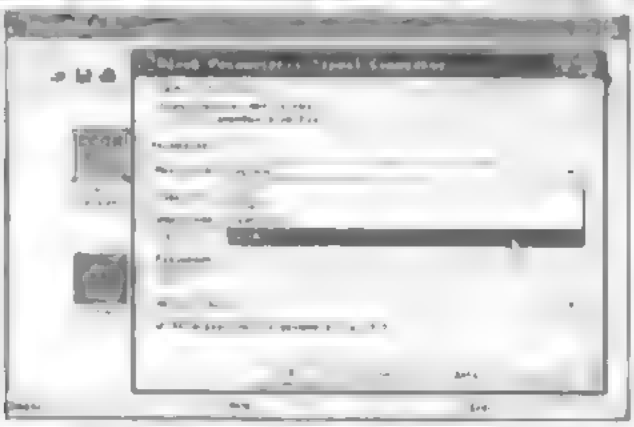


图 8.52 添加"Signal Generator"模块

step 2 [illegible] 模块。[illegible] 模块的"Scope"模块，如图 8.53 所示。

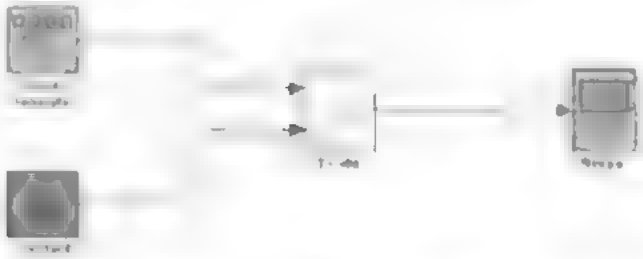


图 8.53 添加接序模块

step 3 查看仿真的结果。单击模块窗口中的"Scope"模块，查看仿真的结果，得到的结果如图 8.54 所示。

图 8.54 查看仿真结果

[illegible]

8.3 Simulink 的基本操作

[illegible]

[illegible]

8.3.1 Simulink 模型的工作原理

[illegible]

[illegible]

[illegible]

图 8.55 代表定义状态的系统模块的图形原理。

图 8.55 模块的图形化形式

[illegible]

[illegible]

[illegible]

[illegible]

- 计算模型参数的表达式并确定它们的数值
- 确定信号的属性
- 传递信号属性并确定没有定义的信号属性
- 对系统模块进行优化
- 展开模型的继承关系
- 确定模型运行的优先级
- 确定模块的采样时间。

[illegible]

每个模块的信息。

仿真过程主要分为两个阶段，第一个阶段是模块初始化阶段，在"Ctrl+D"之后，主要是一个初始化系统的状态和输出。第二个阶段是仿真运行阶段，在初始化之后，[illegible]在每步仿真中调用一次，用于在每一个时间步计算模块的新输入、输出和状态。

8.3.2 模块的操作

模块是构建Simulink模型的基础工具，用户可以在空白文件中通过拖拽和复制的方式将模块放置在窗口中，从而建立各种动态系统。因此，有必要熟悉模块的操作方法，才能顺利地进行建立和编辑系统。

在详细介绍如何操作模块之前，首先提醒一个使用技巧：使用"鼠标指针"工具，可以查看模块的属性和参数，也可以设置属性，当使用鼠标在模块上方移动时，Simulink会显示关于该模块的基本信息，[illegible]查看模块属性，可以通过模块菜单中的"View"菜单选项下的"Block Data Tips Options"命令来设置显示的位置，下面举例详细说明。

例 8.6 设置模块显示的属性选项。

step 1 打开前面保存的"Sim4"文件，然后选择菜单栏中的"View"→"Block Data Tips Options"→"Parameter Names And Values"命令，如图 8.56 所示。

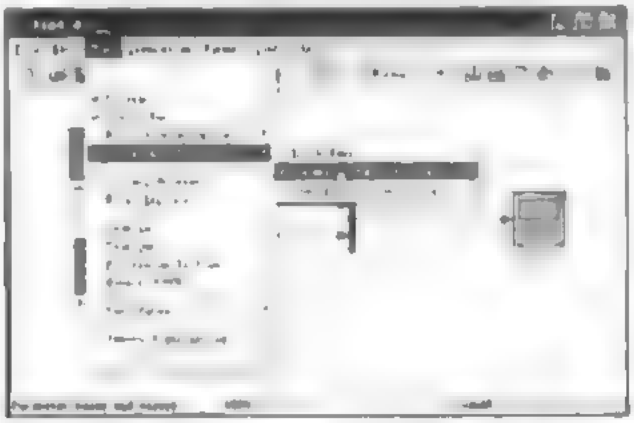


图 8.56 设置模块显示的属性选项

step 2 查看模块的属性。完成上面的设置之后，将鼠标指针在"Sine Wave"模块上，系统显示出该模块的属性和数值，如图 8.57 所示。

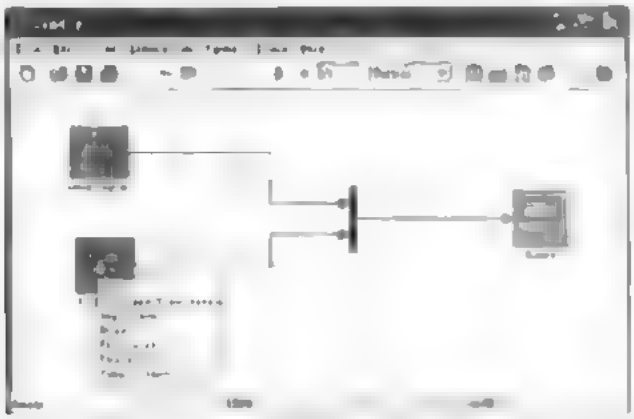


图 8.57 显示模块的属性和数值

Step 3 [illegible]，如图8.58所示。

图8.58 查看新文件的显示属性

[illegible]

[illegible]方法请用户自行尝试。

8.3.3 复制和移动模块

[illegible]和Paste菜单选项等，这里不重复介绍。

[illegible]

旋转模块

[illegible]下的"Flip Block"和"Rotate Block"子菜单选项。

- 选择"Flip Block"菜单命令可以将模块旋转180°。
- 选择"Rotate Block"命令可以将模块顺时针旋转90°。

[illegible]

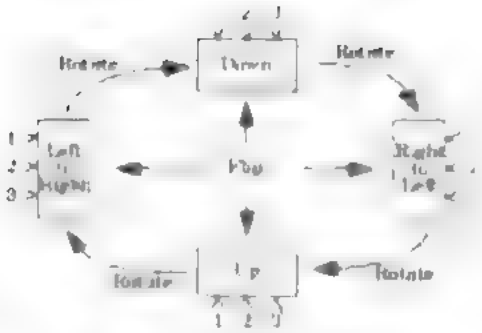


图 8.59 旋转模块的效果演示图

[illegible]

8.3.4 显示模块的属性数值

[illegible]

例 8.7 在模块的下方显示模块的属性数值。

Step 1 [illegible] 模块，单击鼠标右键，在弹出的快捷菜单中选择 "Block Properties" 菜单选项，打开该模块的属性对话框，如图 8.60 所示。

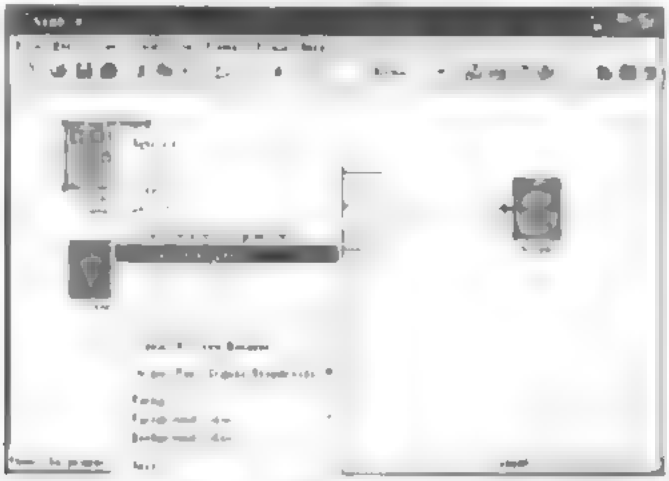


图 8.60 打开模块属性对话框

Step 2 查看模块属性对话框，[illegible]，系统显示对应模块属性的对话框，如图 8.61 所示。

Step 3 设置显示属性的数值。选择对话框中的 "Block Annotation" 选项卡，然后在 "Enter text and tokens for annotation" 选项下面输入文本 "Amplitude=%<amplitude>, Frequency = %<frequency>"，单击 "Apply" 按钮，如图 8.62 所示。

图 8.61　默认的模块属性对话框

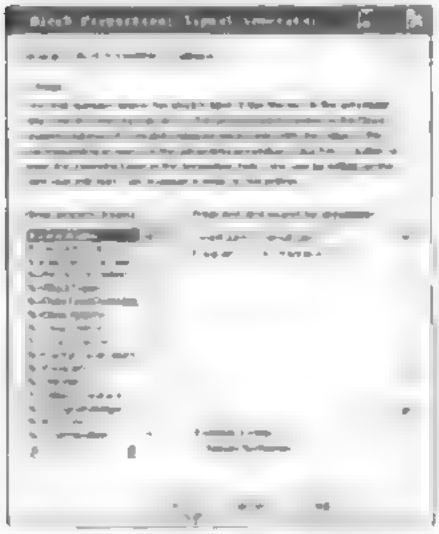


图 8.62　设置显示属性的选项

Step 5　[illegible]，单击"OK"按钮，关闭模块属性对话框，然后查看设置完成后的模块，如图 8.63 所示。

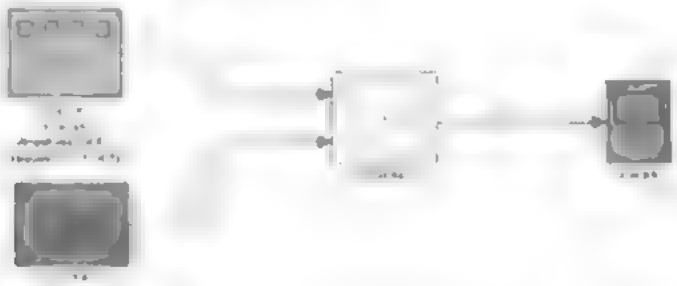


图 8.63　设置完成后的模块

说明　从上面的步骤中可以看出，如设置了在模块下方显示模块数值选项后，在模块的下方将会显示模块属性对话框的数值。[illegible]

8.3.5 添加模块的阴影效果

[illegible]

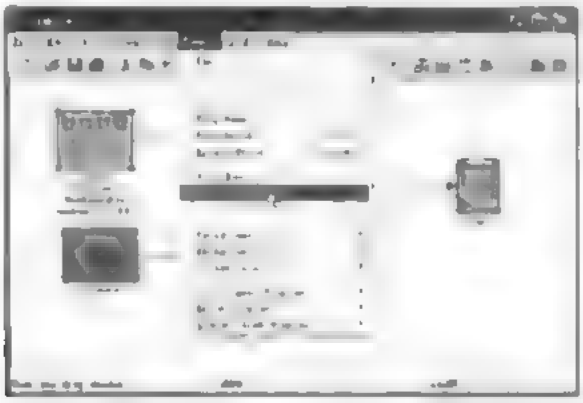


图 8.64 添加模块的阴影效果

[illegible]

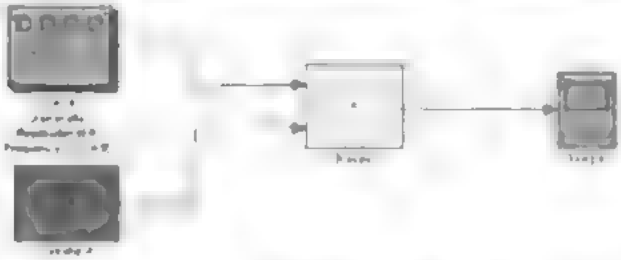


图 8.65 添加阴影效果后的模块

提示

[illegible] Drop Shadow" 变成 "Hide Drop Shadow" 菜单选项。

8.3.6 操作模块名称

[illegible]

图 8.66 模块名称的默认位置

提示

Simulink 全路径名称 "[illegible]" 中的 "[illegible]" 表示在 vdp 模型下的 Mu 模块。因此，在指定模块名时时，需要使用斜杠 "/"，以制 Simulink 全路径时应的名称以分隔路径。

[illegible]，[illegible]，[illegible]，

图 8.68 设置显示模块运行的输出结果

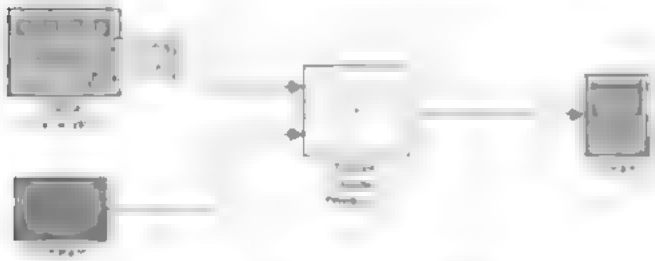


图 8.69 查看用户设置的结果

Step 3 [illegible] Simulink Block Output Display [illegible] 属性，如图 8.70 所示。

图 8.70 设置显示选项

[illegible]

8.3.8 连接线的分支

[illegible] 模块中的连接线设置各种属性。下面详细介绍。

[illegible]

[illegible]

[illegible]

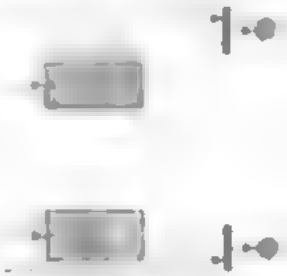


图 8.71　Simulink 中的分支结构

[illegible]

[illegible]

8.3.9 移动连接线的节点

[illegible]

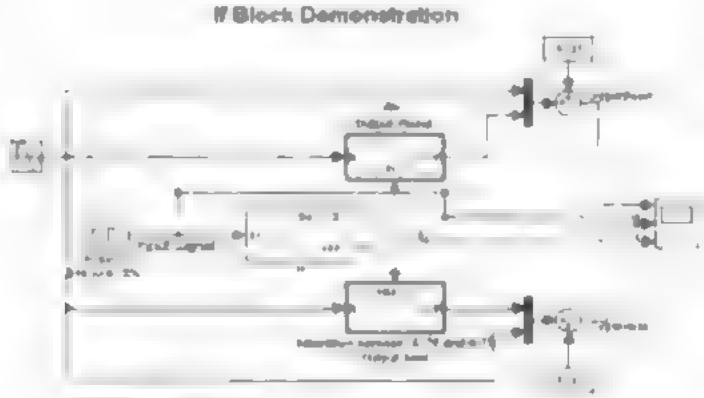


图 8.72　移动连接线的节点位置

[illegible]

8.3.10 彩色显示信号线

在Simulink所创建的高能系统中，允许用户采用多种采样频率，如果希望了解各不同采样频率的模块及信号线，可以为各个采样频率设置不同的颜色。这样，就可以方便地查看系统采样频率。以MATLAB自带的"multirate.mdl"文件为例，说明如何使用彩色显示信号线，如图8.73所示。

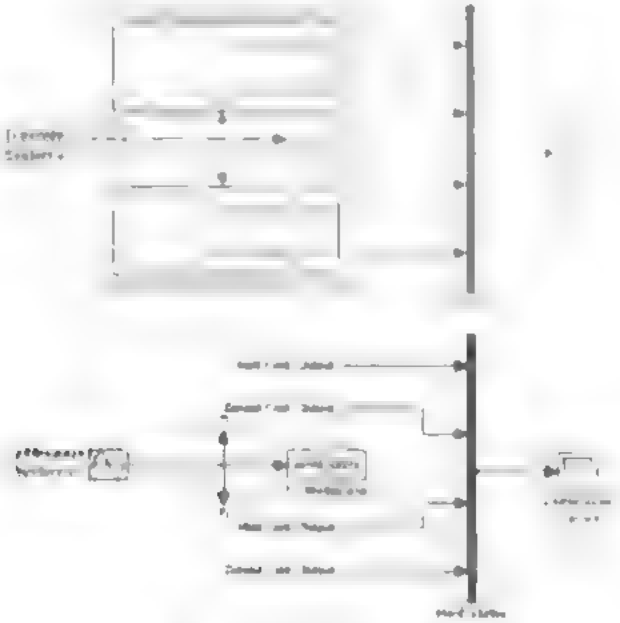


图8.73 彩色显示信号线

在上面的窗口中，选择菜单栏中的"Format"→"Port/Signal Display"→"Sample Time Color"命令后，Simulink自动将采用不同设置采样频率的模块，以不同的颜色显示。

8.3.11 添加信号线标识

现在整个模型中添加了很多的模块，对于信号线，如果为其他用户使用和理解信号线的信息，可以为该信号线添加标识。添加信号线标识的方法非常简单，只需双击信号线，系统会弹出一个带有光标的文本编辑框，在其中输入文本，作为该信号线的标识。为了加深对标识的直观了解，以MATLAB自带的"busdemo.mdl"文件为例，由户可以查看该模型中的信号线标识，如图8.74所示。

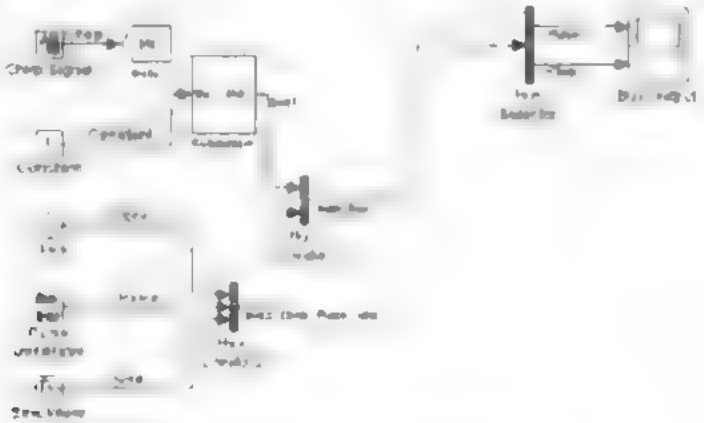


图8.74 添加信号线标识

[illegible]

8.3.12 设置连接线的属性

[illegible]的属性对话框。

[illegible]对话框，如图 8.75 所示。

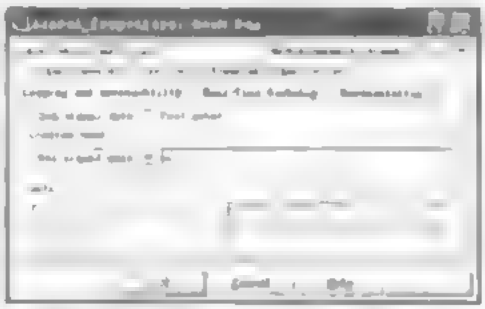


图 8.75 设置连接线的属性

[illegible]

8.4 Simulink 的信号

[illegible]

[illegible]

8.4.1 创建信号

[illegible]节将首先介绍关于信号的基础知识，读者可以对信号有个直观的了解。

[illegible] 模块库中的信号。

[illegible]

8.4.2 添加信号标签

[illegible] 完成名称的修改。

[illegible] 名称，如图 8.76 所示。

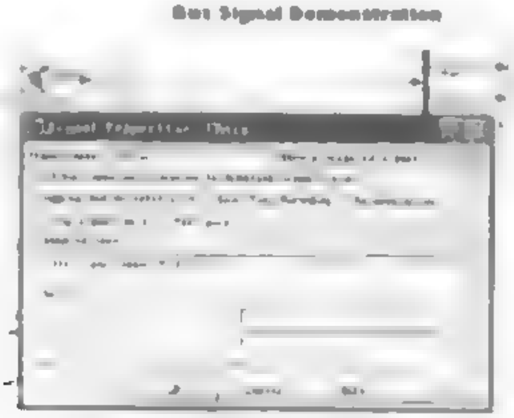


图 8.76 添加信号标签

8.4.3 显示信号数值

[illegible]

[illegible]

8.4.4 复数信号

[illegible]

Simulink中通常使用下面的方法来处理复数信号的模型

- [illegible]
- [illegible]
- [illegible]

例8.9　在Simulink中处理复数信号。

STEP 1　[illegible]

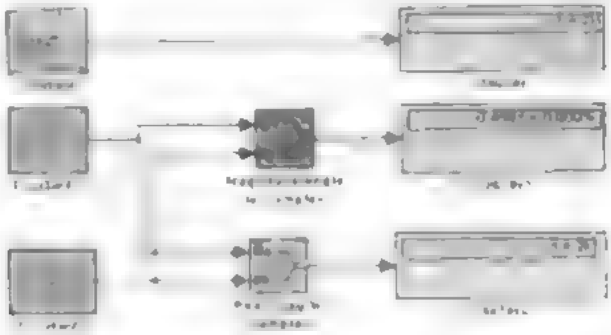


图8.77　使用不同的方法来创建复数信号

在复数的处理过程中，经常要与复数的各个分数，可以使用这种复数分解的处理复数在Magnitude-Angle to Complex模块中，[illegible]

STEP 2　[illegible]

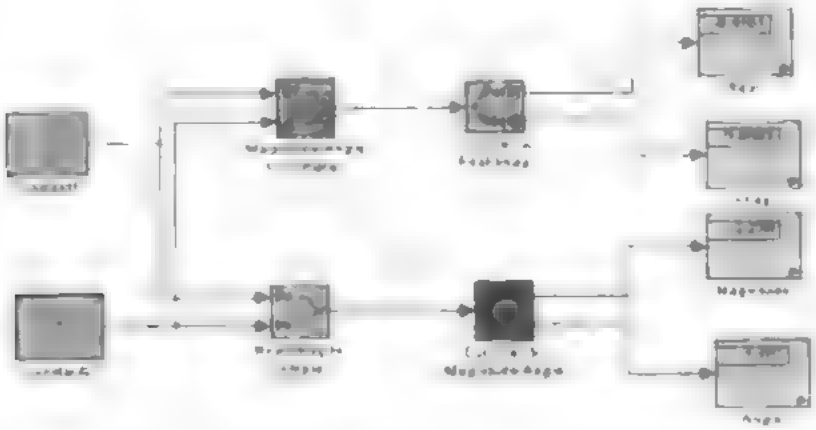


图8.78　转换为实数信号

8.4.5 虚拟信号

虚拟信号是使用多路复用器表示单信号的一个信号。在Simulink中多路复用模块，例如Bus Creator、Inport和Outport block等，[illegible]。虚拟信号[illegible]可定义多维数值实体，与有数学或者逻辑上的作用。与Simulink在进行系统仿真时[illegible]或者更新模型时，系统使用信号端口直接连接虚拟信号的表示信号实体，[illegible]整个系统的[illegible]上面的信号实体对象。下面使用一个简单的实例来说明虚拟信号的概念。

例8.10 使用一个简单的实例来表示虚拟信号的概念。

step 1 打开空白模型编辑界面，在编辑界面中添加"常数"模块、添加Bus Creator、Bus Selector等虚拟信号模块和增益模块，最后添加"Display"模块，如图8.79所示。

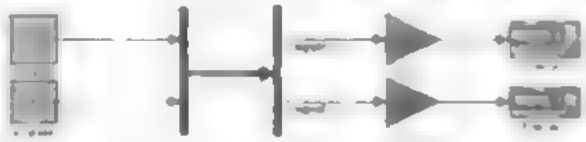


图8.79 添加虚拟信号模块

在复制模块中，添加了两个虚拟信号模块，得到的结果信号有两个增益模块。对，分别将这个参数增益，修改为1，两个增益数值，也就是说，图8.79所示的模块系统效果完全相同。

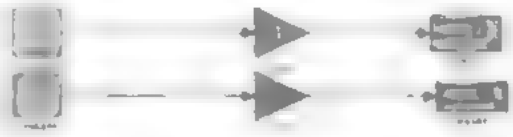


图8.80 清除虚拟信号后的系统

step 2 修改信号名称。在主菜单栏中选"Bus Creator"模块，打开对话框，在其中修改信号的名称，如图8.81所示。

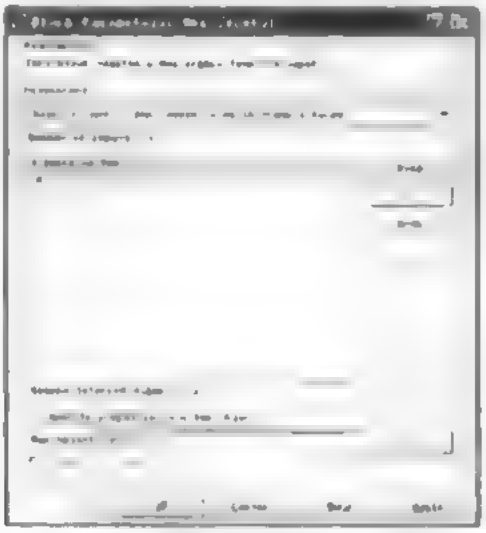


图8.81 修改信号的名称

说明 [illegible]

step 3 [illegible] 中修改信号输出的次序。如图 8.82 所示。

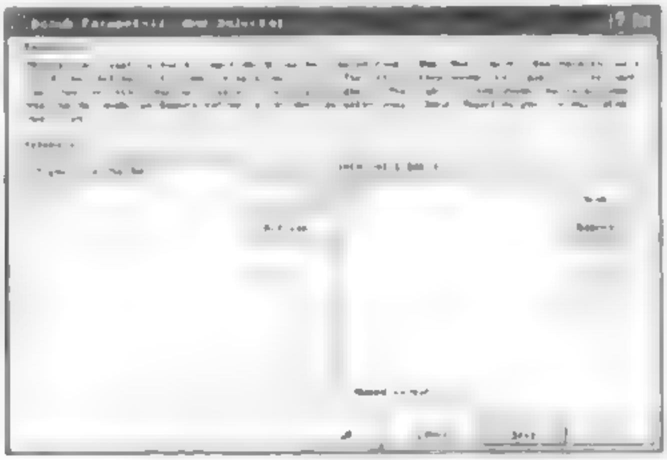


图 8.82 修改信号输出的次序

step 4 [illegible] 新运行该仿真系统，得到的结果如图 8.83 所示。

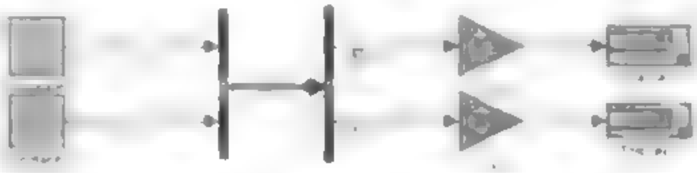


图 8.83 重新运行系统

[illegible]

提示 在 [illegible] "Bus Creator" [illegible]

8.4.6 控制信号

[illegible] 来说明控制信号的使用方法。

例 8.11 [illegible]

step 1 [illegible]

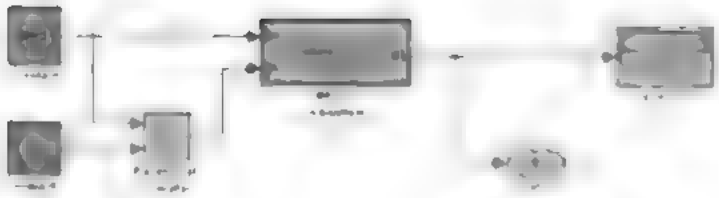


图 8.84 添加 While 子系统

[illegible]

Step 2 设置子系统上模块的属性。"Relational Operator"模块[illegible]，[illegible]模块对应的参数属性，如图 8.85 所示。

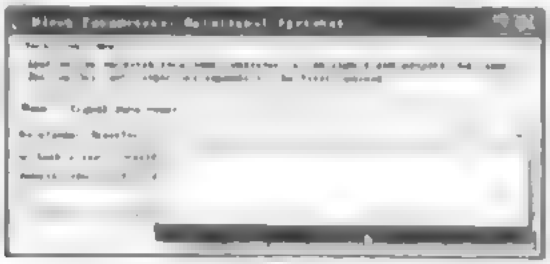


图 8.85 设置关系表达式

[illegible]

Step 3 [illegible]模块，如图 8.86 所示。

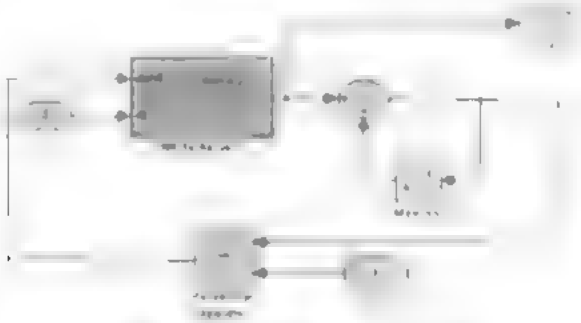


图 8.86 编写子系统的模块

Step 4 设置"While Iterator"模块的属性。[illegible]

[illegible]

Step 5 设置"Memory"模块的属性。[illegible]设置属性，如图 8.88 所示。

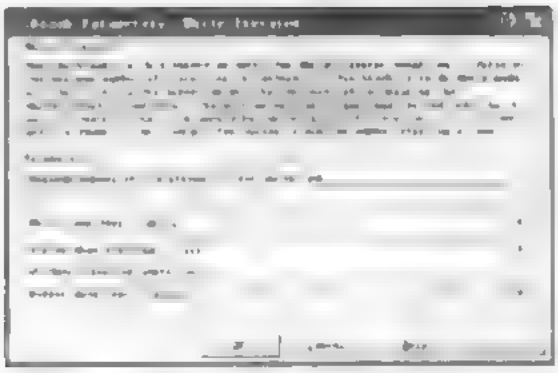


图8.87 设置"While Iterator"模块的属性

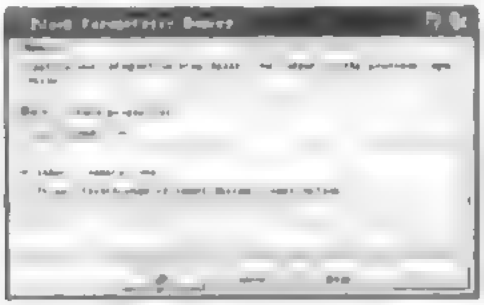


图8.88 设置"Memory"模块的属性

Step 6 [illegible]

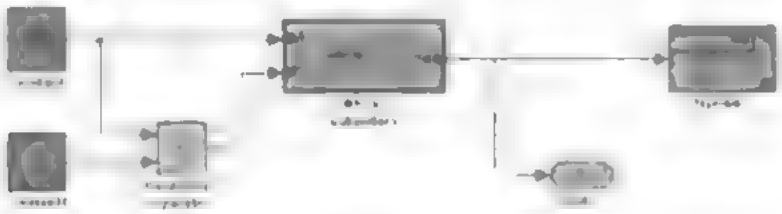


图8.89 得出仿真结果

[illegible] $1+2+3+\cdots+n=\frac{n(n+1)}{2}$，[illegible] $1+2+3+\cdots+13=\frac{13\times(13+1)}{2}=91$ [illegible]

Step 7 [illegible]

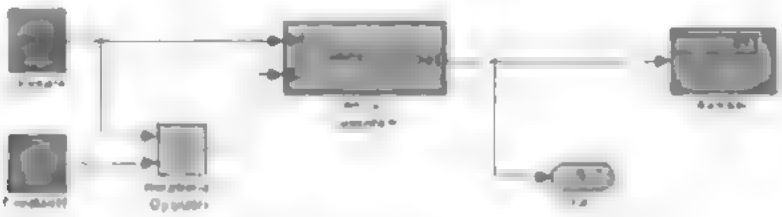


图8.90 修改程序条件

熟悉编程的读者也许会明白，上面仿真系统[illegible]

```
max_sum = 100;
sum = 0;
iteration_number = 0;
cond = (max_sum > 0);
while (cond != 0) {
    iteration_number = iteration_number + 1;
    sum = sum + iteration_number;
    if (sum > max_sum OR iteration_number > max_iterations)
        cond = 0;
```

从上面的循环过程[illegible]，控制信号[illegible]，[illegible]方便地表示各种程序功能。

[illegible]

8.4.7 信号总线（Signal Buses）

在Simulink中，信号总线[illegible]信号。

在Simulink中，[illegible]Bus Creator模块[illegible]Bus Selector模块来接收信号总线。在Simulink的模块库中[illegible]Sinks'模块库中[illegible]所示。

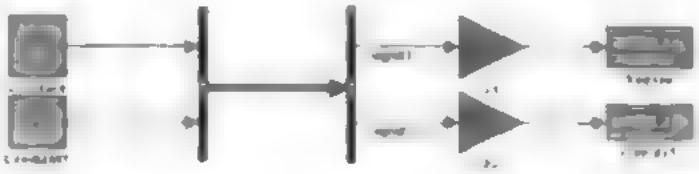


图8.91 信号总线对象

由于信号总线的输入信号比较复杂，如果想查看[illegible]，[illegible]信号总线对象，然后选择'Format'→'Port/Signal Displays'→'Signal Dimensions'命令，就可以查看信号总线的组成信号个数，如图8.92所示。

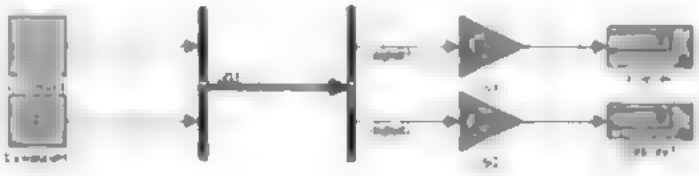


图8.92 显示信号总线的信号个数

[illegible]

[illegible]

[illegible]

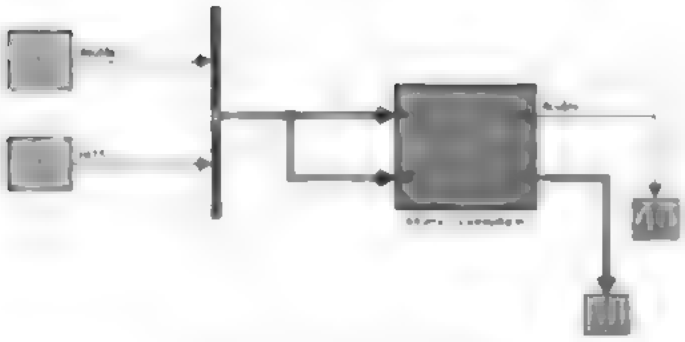


图8.93 添加程序系统的模块

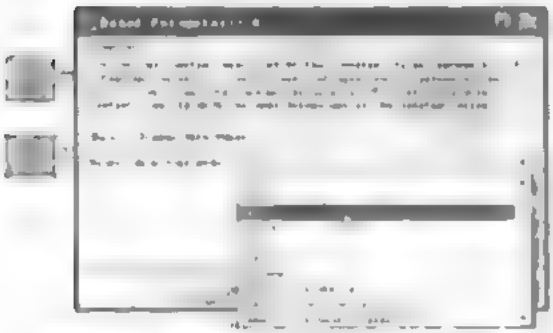


图8.94 修改模块输出变量的属性

Step 3 建立子系统模块。[illegible]，以系统模块编辑窗口，在其中建立对应的子系统模块，如图8.95所示。

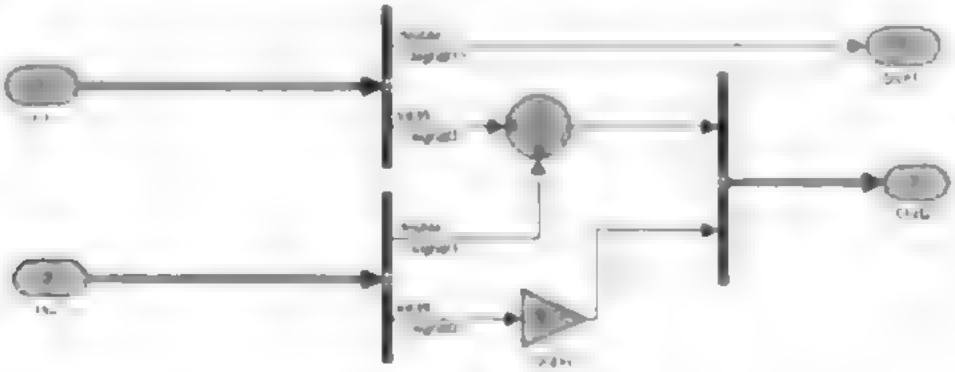


图8.95 创建子系统模块

在上面子模块中，将系统[illegible]输入[illegible]，创建新的数据。之所以设计这样的模块结构，是为了在[illegible]

Step 4 保存模型，[illegible]，如[illegible]，则可将系统提示如图8.96所示。

提示：在创建系统模块[illegible]，系统中[illegible]输入[illegible]的函数，[illegible]模块[illegible]数据传递，自动[illegible]仿真过程。

Step 5 单击对话框的"[illegible]"按钮，返回到子系统模块视图中，在其中添加信号转换模块，如图8.97所示。

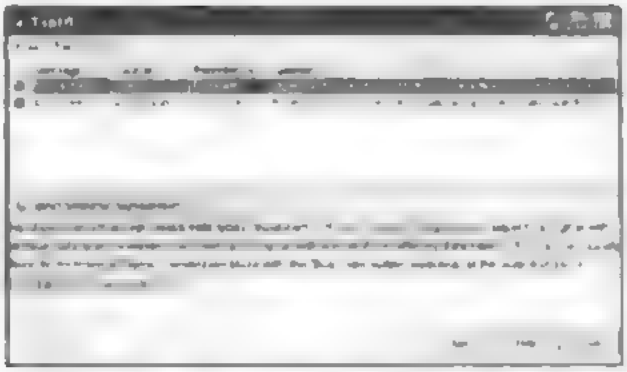


图 8.96　系统提示的错误信息

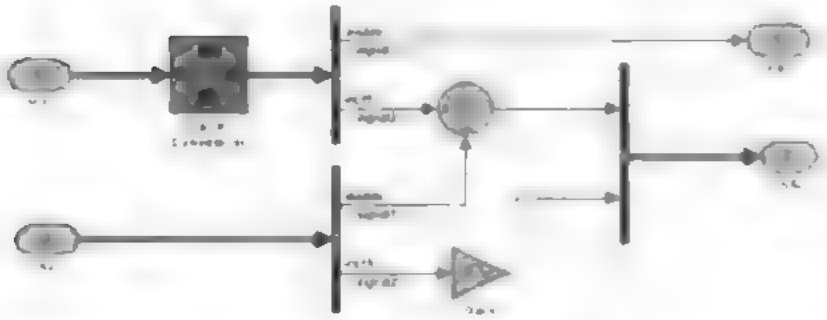


图 8.97　添加信号转换模块

step 6 设置"Signal Conversion"模块的属性。双击该模块，打开其属性对话框，将输入变量的数据类型选择为"Bus copy"，如图 8.98 所示。

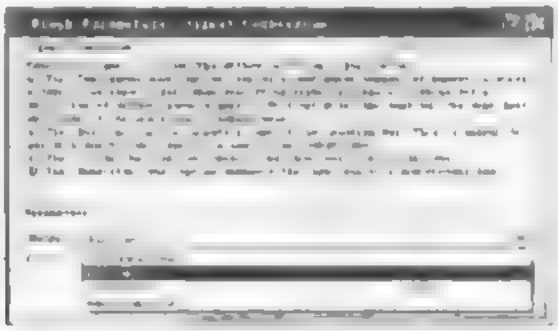


图 8.98　设置信号转换模块的属性

step 7 设置"Gain"模块的"Fixed-Point"属性。选择上面的"Gain"模块，然后选择"Tools"→"Fixed-Point Setting"命令，打开"Fixed-Point Settings"对话框，设置其属性选项，如图 8.99 所示。

在上面的对话框中，将模块的"Output Data Type Override"属性设置为"True"，将模块的"Output Type Override"属性设置为"Inherit: Same as input"，设置完毕后，单击"确定"按钮关闭对话框，使 Simulink 可以正常运行该系统。

说明　上面的实例中没有涉及变量的转换问题，不需要在工作空间定义变量的情况。关于上面的属性设置问题，将会在接下来的章节中进行介绍。

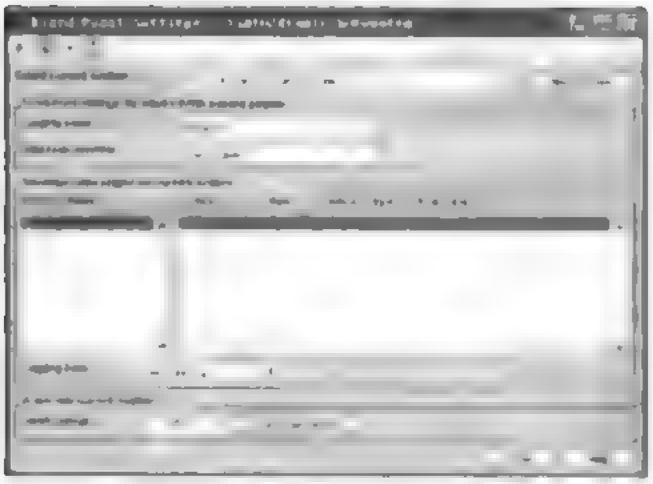


图 8.99 设置"Fixed-Point"属性

8.4.8 信号组

[illegible]模块，将其拖放到模块编辑器中，创建一个对象[illegible]，创建系统模块如图 8.100 所示。

图 8.100 创建的系统模块

[illegible]模块，[illegible]

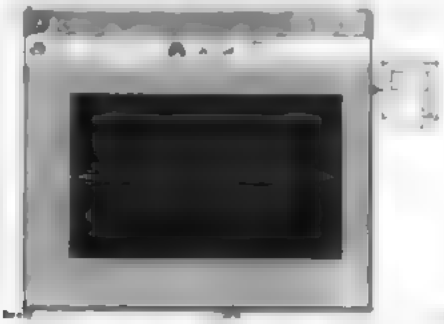


图 8.101 查看默认的信号组

如图 8.101 所示[illegible]，在默认情况下 Simulink 提供的信号组[illegible]冲突为单个信号。

[illegible]信号编辑器的"Signal Builder"[illegible]编辑器，[illegible]信号的属性，选择[illegible]等，[illegible]系统中的"Signal Builder"模块，可以打开对应的属性编辑器，如图 8.102 所示。

在多对话模块，可以利用编辑器的"Signal Builder"模块[illegible]。其中[illegible]重要的控件。

◆ "Group"面板：[illegible]

图 8.102 信号组属性编辑器

◆ **信号坐标轴：**[illegible]

◆ **信号列表：**[illegible]

[illegible]

例 8.13 显示如何编辑信号组对象。

step 1 [illegible]

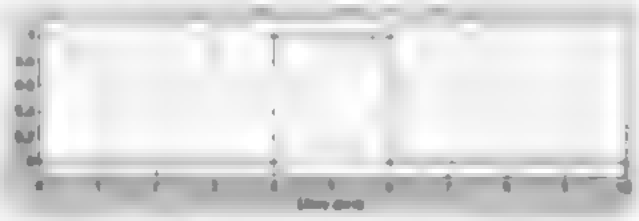


图 8.103 选中波形对象

[illegible]

step 2 [illegible]

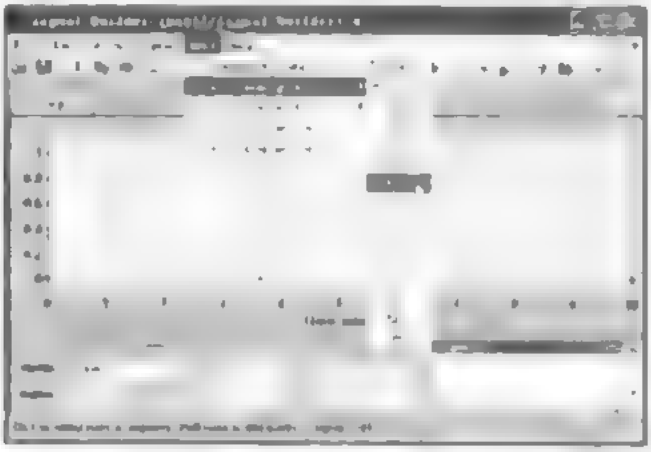


图 8.104 设置振幅坐标轴的网格属性

[illegible]

step 3 [illegible]

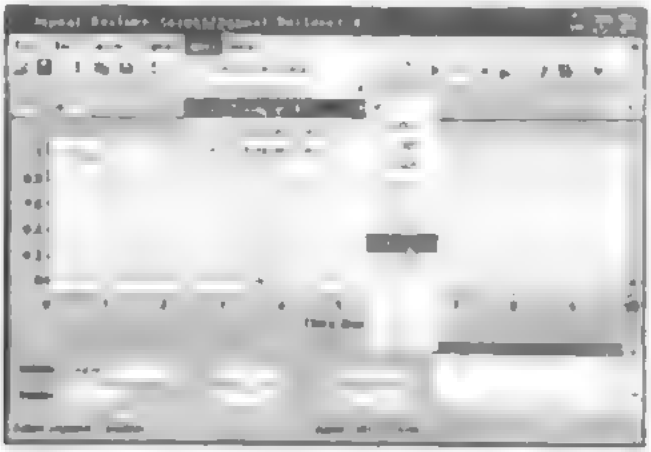


图 8.105 设置时间轴的网格属性

[illegible]

step 4 [illegible]

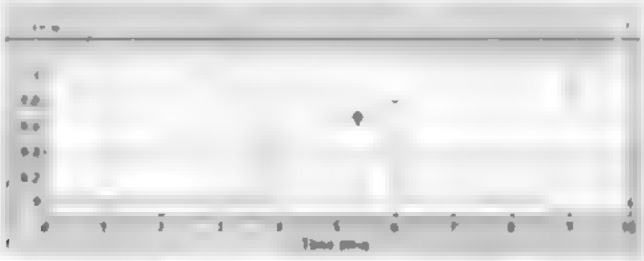


图 8.106　上下移动波形片段

step 5 [illegible]所示。

图 8.107　左右移动波形片段

[illegible]的数值相差为 4.1，这是最小的移动单位。

step 6 [illegible]中输入该数据点的坐标数值，如图 8.108 所示。

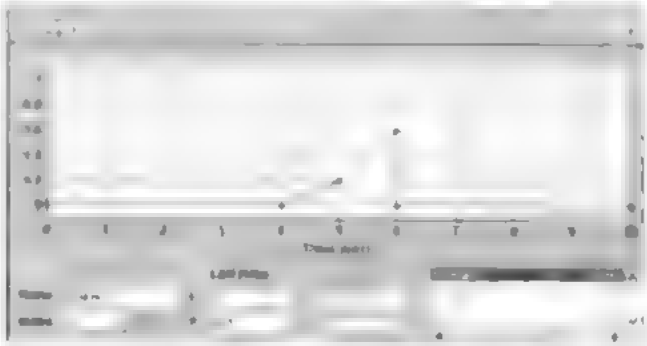


图 8.108　修改数据点的坐标数值

说明 [illegible]

step 7 [illegible]

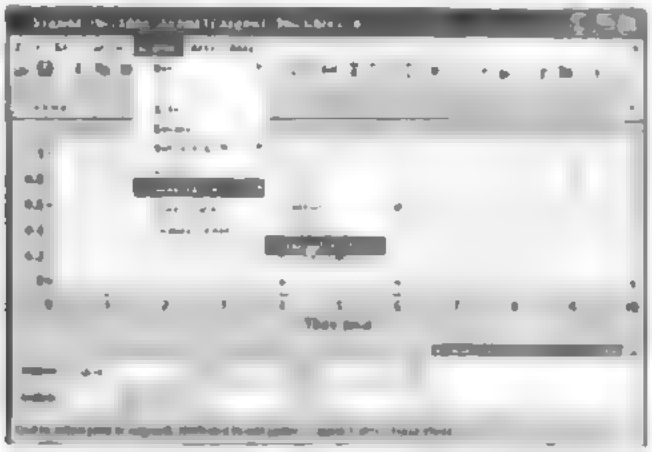


图 8.109 修改波形图形的线型

[illegible]

step 8 [illegible]

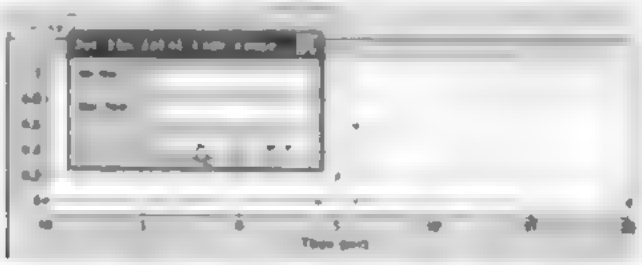


图 8.110 设置波形的时间范围

step 9 [illegible]

图 8.111 将波形输出到工作空间

step 10 [illegible]

图 8.112　添加新的信号对象

step 11 [illegible]以在波编辑器中添加"Signal 2"信号，如图 8.113 所示。

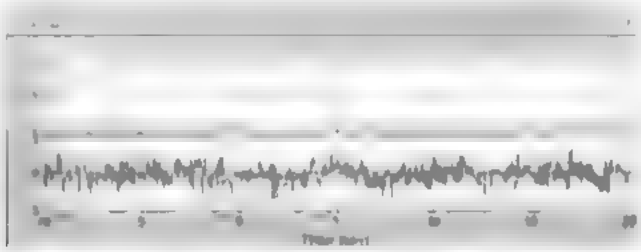


图 8.113　添加后的信号对象

step 12 [illegible]"Delete"命令，删除源象的信号对象，如图 8.114 所示。

图 8.114　删除源象的信号

step 13 [illegible]所示的结果。

step 14 [illegible]Options"对话框，在其中设置仿真的参数，如图 8.116 所示。

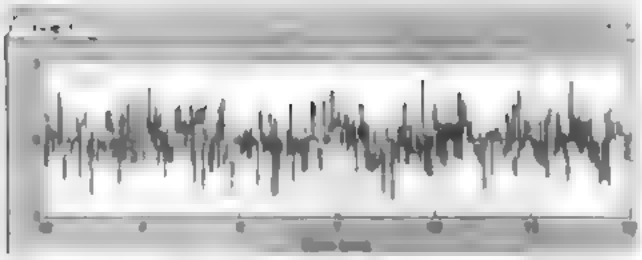


图 8.115 删除后的信号

图 8.116 设置仿真参数

单击对话框中，将“Sample time”设置为[]，表示Simulink根据结果信号的类型，采用默认系统的默认设置。单击对话框的“OK”按钮，完成设置。

step 15 查看仿真结果。返回模型窗口，单击“开始仿真”按钮，并双击“Scope”模块，查看仿真结果如图 8.117 所示。

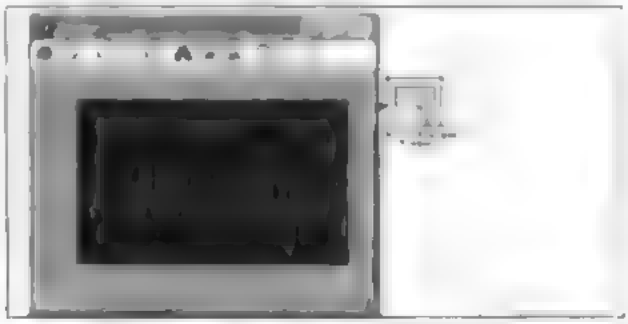


图 8.117 查看连续的仿真结果

step 16 修改仿真参数。单击“Signal Builder”模块，选择“File”→“Simulation options”命令，打开“Simulation options”对话框，修改仿真参数，直接点击“OK”。

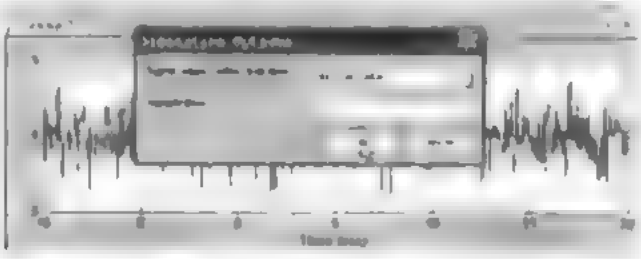


图 8.118 修改仿真参数

step 17 查看仿真结果。返回模型窗口，单击“开始仿真”按钮，并双击“Scope”模块，查看仿真结果如图 8.119 所示。

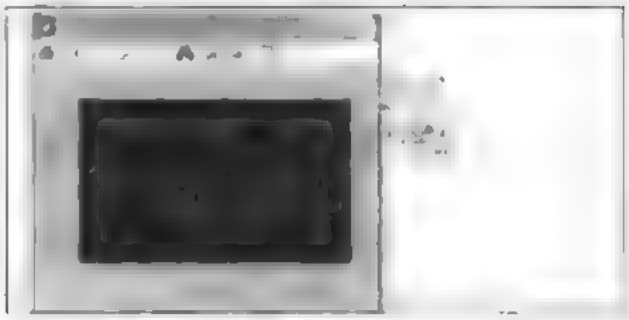


图8.119 显示离散的仿真结果

[illegible]

8.4.9 使用自定义信号源

[illegible]

例8.14 [illegible] $y=e^{-t}\cos(\frac{t}{2})$ [illegible]

Step 1 [illegible] 的程序代码

[illegible]

在输入上面的程序代码后，将其保存为"mysource.m"文件。

Step 2 [illegible]

图8.120 添加程序模块

Step 3 [illegible] 模块，打开其参数对话框。在"Data"数值框中输入"Ts"，如图8.121所示。

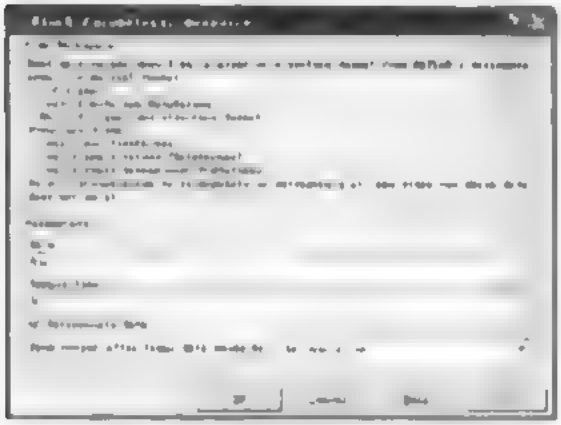


图 8.121 设置模块的参数

step 4 在 MATLAB 的命令窗口中输入下面的命令

```
>> Ts=mysource;
```

上面程序代码在 MATLAB 的工作空间中添加 Ts 变量。

step 5 单击全模型窗口中，设置仿真时间为"10"，单击"开始仿真"按钮，进行系统的仿真，并双击"Scope"模块，查看仿真结果，如图 8.122 所示。

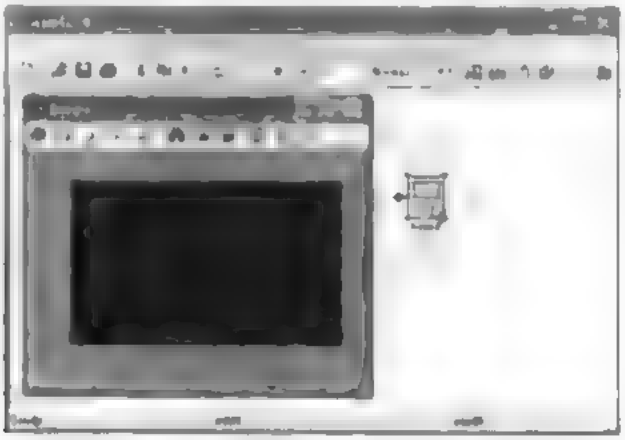


图 8.122 查看仿真的结果

step 6 在命令窗口中查看仿真结果，为了确定仿真的结果，可以在 MATLAB 的命令窗口中绘制相应的函数图形。在命令窗口中输入下面的代码

```
>> t=linspace(0,4*pi,100);y=exp(-t/3).*cos(1/2*t);
>>plot(t,y,'r','LineWidth',2)
>>grid on;axis tight
```

step 7 查看图形结果。输入分号，按"Enter"键，得到分别绘制的与 Scope 相同。

8.4.10 信号接收器

在前面的章节中，读者已经多次接触到基于 Simulink 仿真系统的模块结构，其中常用的信号接收器有 Scope、Display 和 Term matrix 等模块，可以根据需要来选择合适的接收器。由于 Scope 示波器模块是最常用的信号接收器模块，所以，在本小节中将以 Scope 模块为基础介绍。

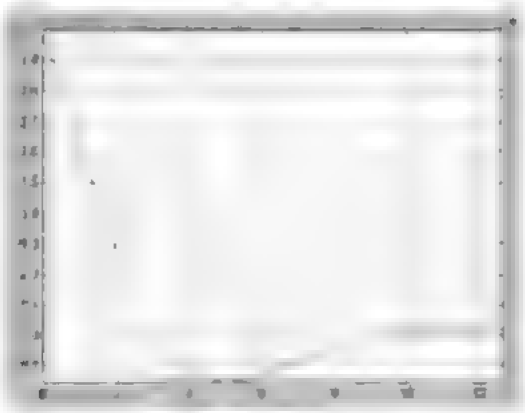


图 8.123 命令窗口的结果

[illegible]

下面将以例 8.14 中的示波器为例，说明如何编辑示波器。

例 8.15 演示如何编辑示波器的属性。

step 1 设置坐标轴属性。[illegible]，单击鼠标右键，在弹出的快捷菜单中选择“Axes Properties”菜单选项，如图 8.124 所示。

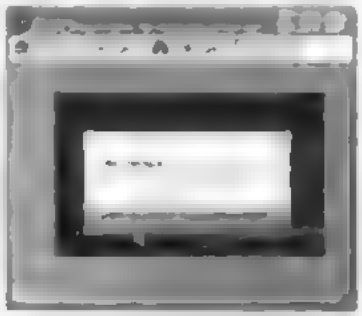


图 8.124 设置示波器的坐标属性

[illegible]，如图 8.125 所示。

图 8.125 设置 Y 坐标轴的上下限

step 2 [illegible]

step 3 设置坐标轴属性。[illegible]

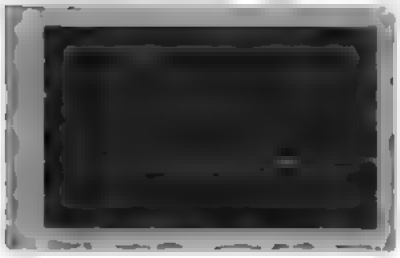


图 8.126 修改后的显示情况

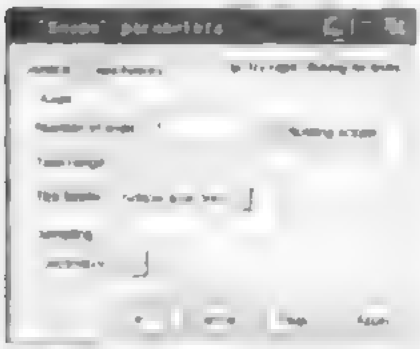


图 8.127 设置横坐标的属性

下面详细介绍上面选项卡的各选项含义。

- **"Time range" 选框：** 该选框设置的是横坐标[illegible]此横坐标轴就是时间轴。"Time range"[illegible]示的是区间[0，10]的波形。如果选择"[illegible]则需要修改其数值，例如在本例中选择[illegible]
- **"Sampling" 下拉菜单：** 该下拉菜单包含两个菜单选项，抽选Decimation和采样时间Sample time。其中，Decimation选项用来设置[illegible]，如果选择n，则每隔n－1个数据点给不显示，默认数值为1。Sample time选[illegible]数据点的采样时间步长。默认数值为0，表示显示连续信号，如果输入的数值[illegible]1，则表示信号显示的方式取决于输入的信号，如果输入的数值大于0，则显示离散信号。

当设置上面的属性后，选择对话框中的[illegible]，在其中设置关于信号数据的属性，如图 8.128 所示。

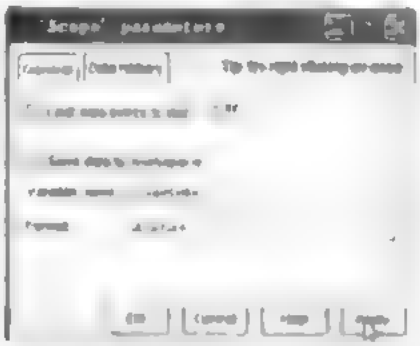


图 8.128 设置信号数据属性

[illegible]

- Limit datapoints to last：[illegible]
- Save data to workspace：[illegible]中的数据送到工作空间中。用户可以自行设定变量名称。

[illegible]

[illegible]

例 8.16 下面如何在 Simulink 中使用波形示波器

Step 1 [illegible]

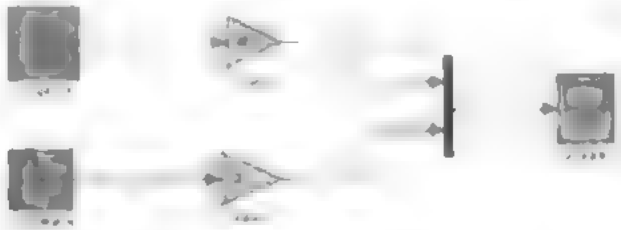


图 8.129 系统模块

Step 2 [illegible]模块，查看仿真结果，如图 8.1[illegible]所示。

Step 3 [illegible]，如图 8.1[illegible]所示。

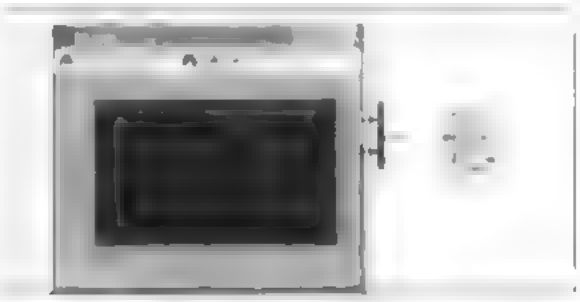


图 8.130 查看仿真的结果

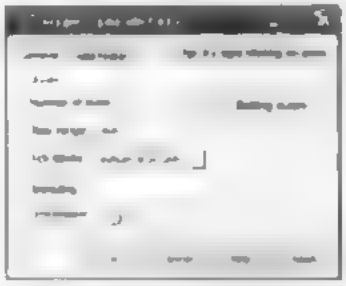


图 8.131 设置坐标轴个数

Step 4 [illegible]模块[illegible]，仿真结果如图 8.132 所示。

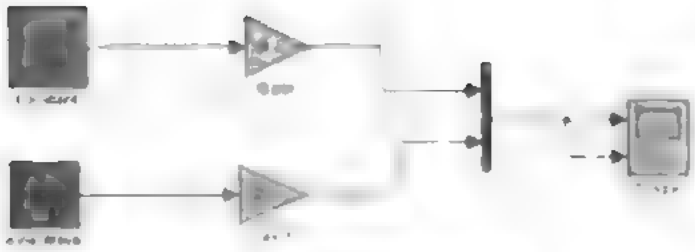


图 8.132 重新连接模块

step 5 重新查看仿真结果。将仿真[illegible]，[illegible]，[illegible]，[illegible]，[illegible]。双击上面的"Scope"模块，查看仿真结果，如图 8.133 所示。

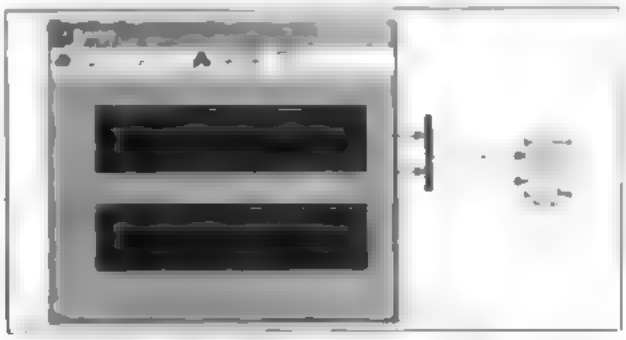


图 8.133 查看新的仿真结果

step 6 将示波器转换为"游离示波器"。[illegible]，[illegible]，[illegible]，[illegible]，将示波器转换为"游离示波器"，如图 8.134 所示。

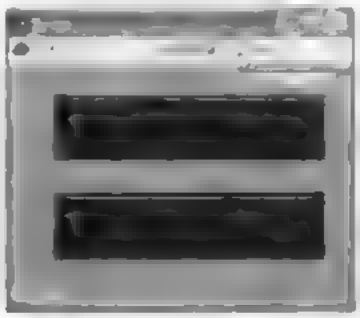


图 8.134 转换为游离示波器

step 7 打开"Signal Selector"的选项。[illegible]，[illegible]，[illegible]，[illegible]，弹出的"快捷菜单"中选择"Signal Selection"选项，如图 [illegible]

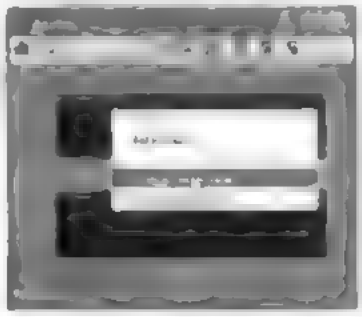


图 8.135 选择"Signal selection"选项

[illegible] 打开 "Signal Selector" 对话框，在其中选择第一个坐标系统的波形，如图8.1 [illegible] 所示。

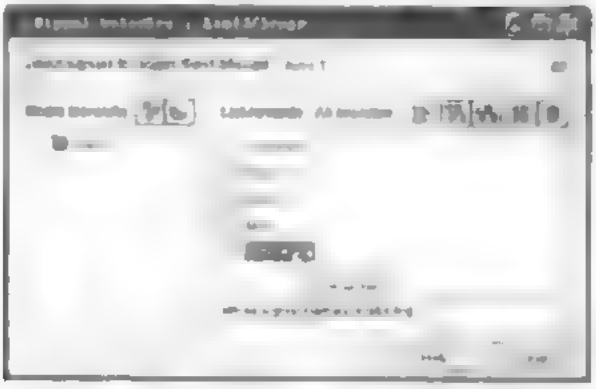


图8.136 选择第一个坐标轴中的波形

step 8 [illegible] 选择 "Selected [illegible]" 选项，如图8.137所示。

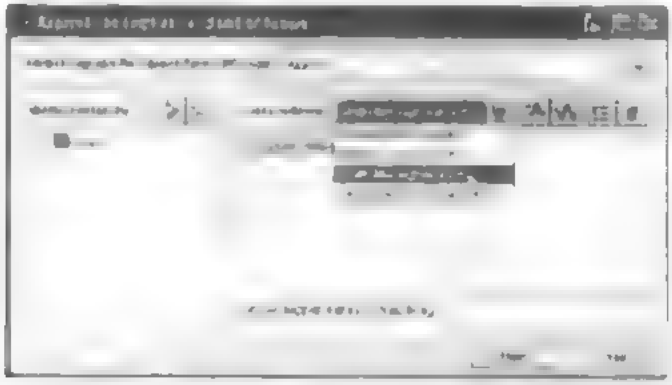


图8.137 改变第二选项

step 9 [illegible] 设置为 "20"，然后单击 "[illegible]" 按钮，进行仿真，并在 [illegible] 选择 "[illegible]" 选项，查看仿真结果，如图8.138所示。

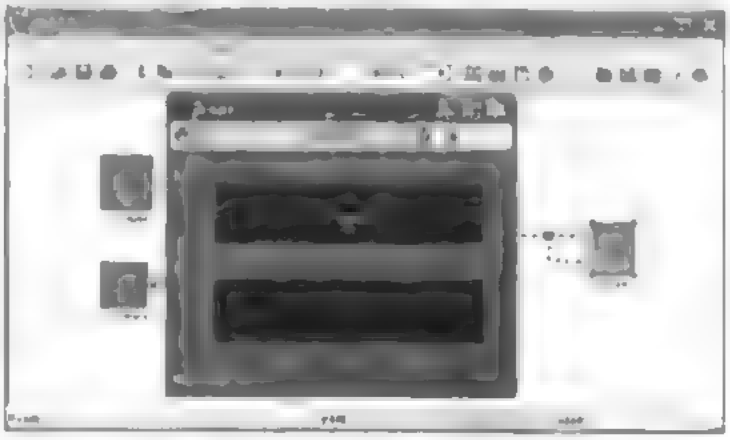


图8.138 新的仿真结果

[illegible]

Step 10 单击下拉菜单，将"Axes"[illegible]Selector"参数对话框如图 8.139 所示。

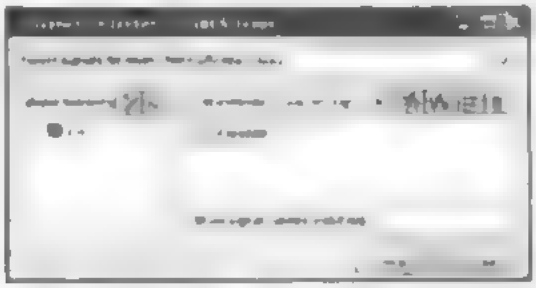


图 8.139 设置"Axes 2"的信号属性

Step 11 查看仿真结果，[illegible]，单击[illegible]仿真，观察仿真结果如图 8.140 所示。

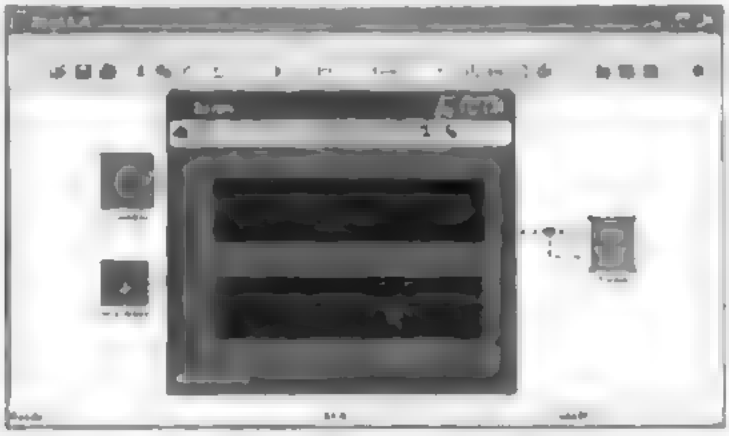


图 8.140 显示新的仿真结果

8.5 Simulink 仿真的设置

根据前面章节的介绍，Simulink 模型在本质上就是一组计算机程序，而这类描述仿真系统的相微分或差分方程，大小指挥[illegible]。Simulink 为[illegible]种数值求解[illegible]方程。[illegible]系统仿真之前，需要根据系统特性和参数设置，对[illegible]在进行仿真之前设置各种仿真参数。

在 Simulink 中，需要设置的仿真参数主要有[illegible]时间、仿真步长、仿真精度、数值积分算法等，还可以设置系统在[illegible]数据。这些设置的仿真数据称为仿真参数，且不同参数都可以在"Configuration Parameters"[illegible]设置。[illegible]各种参数。

8.5.1 设置解算器参数

在设置仿真参数之前，首先需要[illegible]，[illegible]，先将模型窗口中的

"Simulation"、"Configuration Parameters"命令，打开对话框，如图8.141所示。

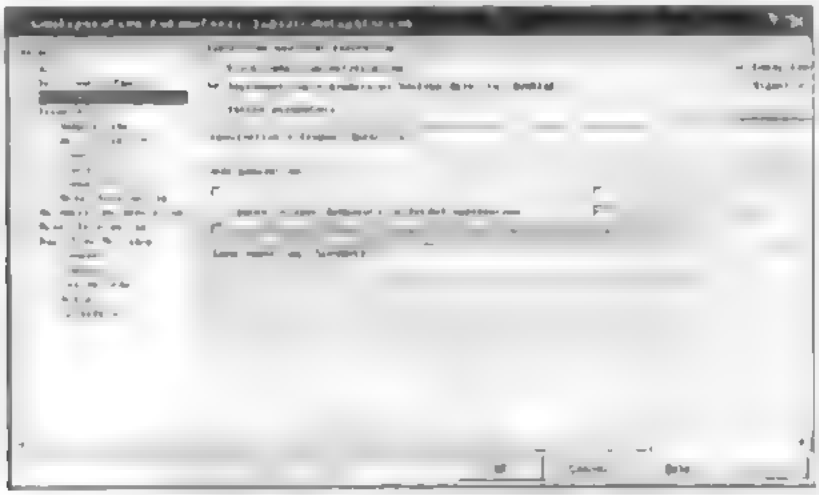


图8.141 仿真参数对话框

在该对话框中设置解算器的参数项，在"Configuration Parameters"对话框的左侧选择"Slover"选项，对应的Slover面板如图8.142所示。

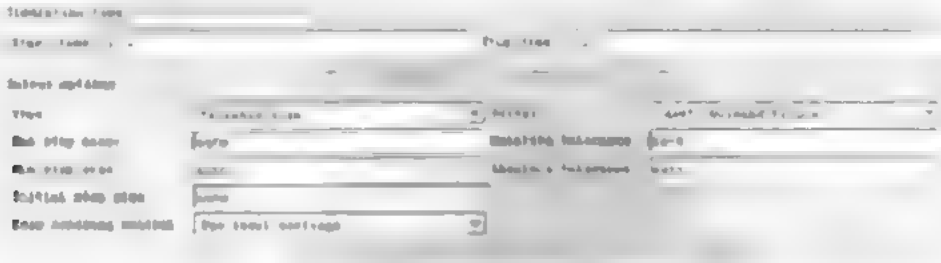


图8.142 解算器参数设置面板

下面将详细介绍上面对话框的各项参数的含义。

- **Simulation time面板**：[illegible]在"Start time"选框中设置新的起始时间，在"Stop time"选框中设置新的终止时间。

[illegible]

- **Solver options面板**：[illegible]Variable-Step解算器属变解算器，默认设置是变步长的ode45。此种解算器在计算下一个仿真时刻的下点，都是在当前仿真时刻上加上一个时间步长。其主要区别在于，固定步长计算器的时间步长是常数，而变步长解算器的时间步长是根据模型动态特性来自动的。

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

表 8.6 Simulink 中的定步长解算器

解算器命令 Solver	使用的方法
ode5	ode45 的定步长的计算形式
ode4	[illegible]
ode3	ode23 的定步长的计算形式
ode2	定步长的 2 阶的 Runge-Kutta 计算方法
ode1	定步长的 Euler 方法
[illegible]	[illegible]

[illegible] Solver 命令，[illegible]

[illegible]

[illegible]

8.5.2 仿真数据的输入输出设置

[illegible] Parameters" 对话框的左侧选择 "Data Import/Export" 选项，在其右侧弹出 "Data Import/Export" 面板，如图 8.143 所示。

[illegible]

[illegible]

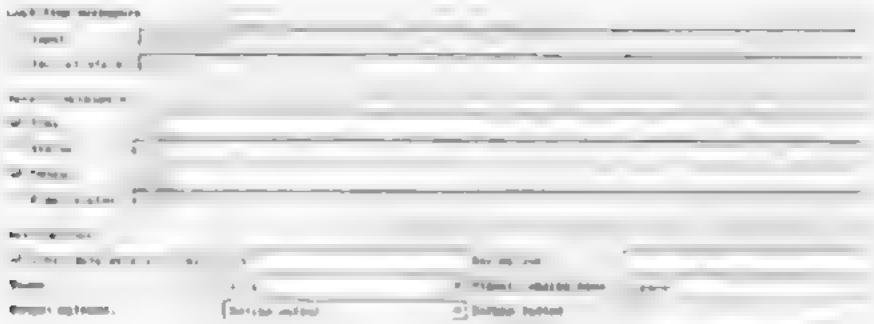


图 8.143 数据输入/输出设置

下面详细介绍该面板各选项的具体含义。

- [illegible] 这些选项框的主要内容如下。
  - **Input 选项框**：[illegible] 则u的第1、2、3…n列分别送往输入模块 In1、In2…In n。
  - **Initial state 选项框**：[illegible] 选项框所填写的变量名称应该是工作空间中所存在的变量。
- [illegible] 框。下面详细介绍这些选项框的具体含义。
  - **Time 选项框**：[illegible] 默认名称为 tout）存放在工作空间中。
  - **States 选项框**：[illegible] 默认情况下为 xout）存放在工作空间中。
  - **Output 选项框**：[illegible]
  - **Final state 选项框**：[illegible] 值，那么这新一轮仿真是前一轮仿真的"继续"。

[illegible]

- [illegible] 各种选项的内容。
  - **Limit rows to last 选项框**：[illegible] 默认数值为 inf。如果输入数据一旦超过设定的数值，则"多余"数据会被自动删除。
  - **Decimation 选项框**：[illegible] 默认数值为 1，即每隔 n－1 个数据点

就会保存一个“解点”。默认数值为1。

- Format选项框：[illegible]

8.5.3 仿真诊断设置

在[illegible]中，提供多种运算[illegible]属性设置，其[illegible]。其中，关于“Slover”的异常诊断属性如图8.144所示。

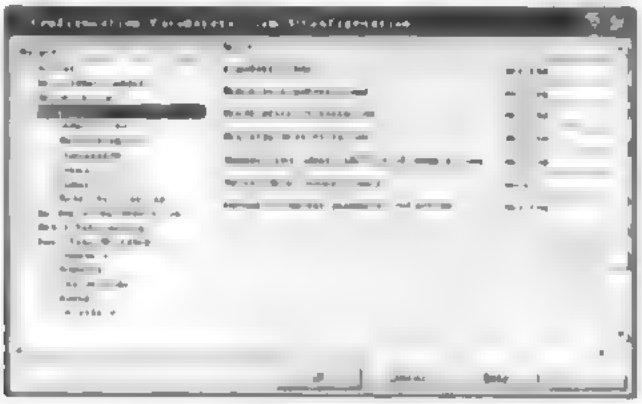


图8.144 关于“Slover”的异常诊断

[illegible]，在[illegible]，关于数据完整性[illegible]，[illegible]

图8.145 关于“Date Integrity”的异常诊断

[illegible]

- None：提示Simulink可以在仿真过程中忽略这种异常。
- Warning：[illegible]
- Error：[illegible]

[illegible]

- Algebraic loop：代数环异常处理，在Simulink[illegible]，[illegible]，[illegible]转换为None。
- Min step size violation：[illegible]

[illegible]

- Unconnected block input：[illegible]
- Unconnected block output：[illegible]
- Consistency checking：[illegible]
- Invalid root Inport/Outport block connection：[illegible]

[illegible]

8.6 Simulink 线性系统建模

[illegible]

8.6.1 线性系统建模简介

[illegible]

例 8.17 [illegible]

Step 1 [illegible]

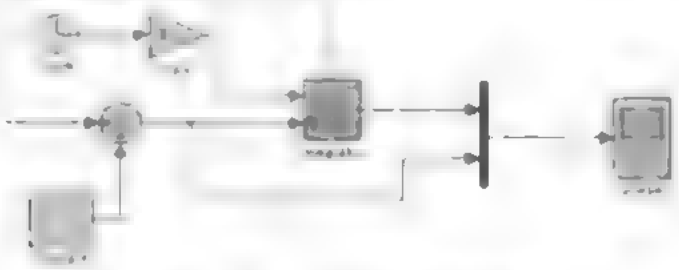


图 8.146 添加基本系统模块

step 2 设置"Sum"模块的属性。双击[illegible]"Sum"模块，[illegible]，在"List of signs"选项中设置修改为"|++"，如图 8.147 所示。

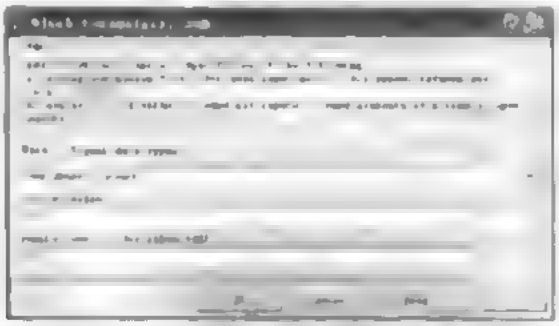


图 8.147 设置"Sum"模块的属性

在 Simulink 中，"Sum"模块在"Math Operations"模块组中[illegible]，[illegible]，其输入[illegible]"|++"，[illegible]，默认情况下其数值为"|++"，表示[illegible]的对应关系。[illegible]，[illegible]

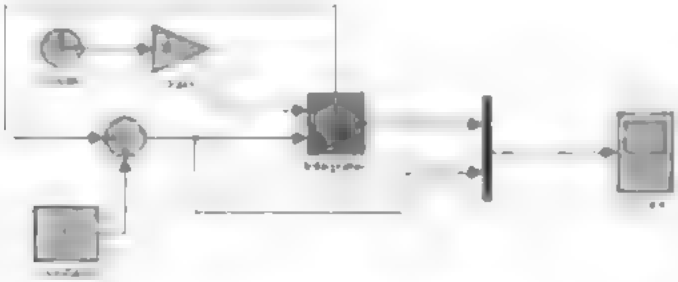


图 8.148 修改后的模块

step 3 设置"Integrator"模块的属性。双击[illegible]"Integrator"模块，[illegible]属性对话框，设置对应的属性，如图 8.149 所示。

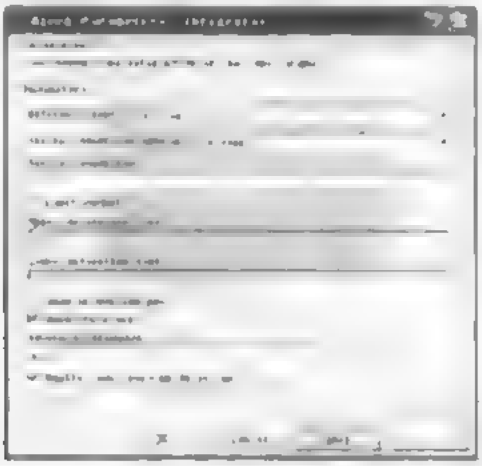


图 8.149 设置"Integrator"模块的属性

在对话框中"External reset"选项中选择"rising"选项，表示[illegible]

[illegible]时，选中对话框中的"Show state port"选项。这样，积[illegible]添加一个输出端口，该端口将会输出关于积分状态的信号。

step 4 [illegible]，得到仿真结果，如图8.150所示。

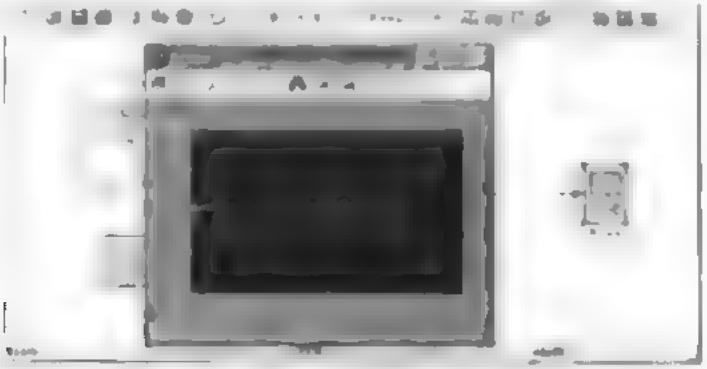


图8.150　仿真得到的结果

[illegible]的结果是比较粗糙的。

step 5 [illegible]，如图8.151所示。

图8.151　设置系统仿真的显示范围

step 6 [illegible]"Apply"按钮，再单击"OK"按钮后，得到新的仿真结果如图8.152所示。

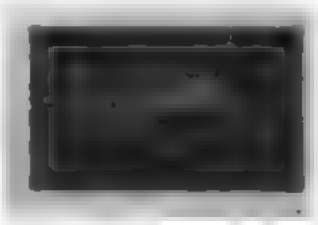


图8.152　新的仿真结果

在本实例完成之后，需要分析该整个系统的基本原理。

step 1 首先，[illegible]，$y(t)=\int_0^t 2t\,dt=t^2$，[illegible]模块"Scope"中。这就是上面仿真系统中的中间线路。

step 2 [illegible]

step 3 [illegible]

step 4 [illegible]

8.6.2 线性系统建模实例

[illegible]

例 8.18 [illegible]

step 1 [illegible]

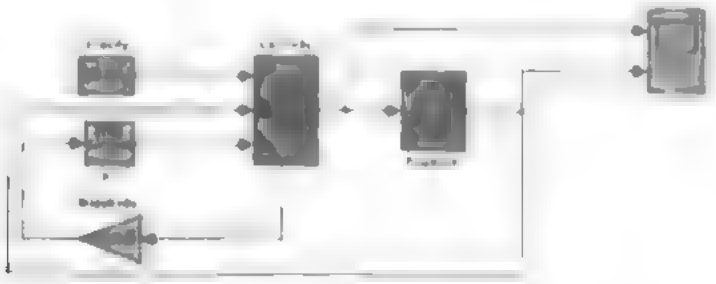


图 8.153 添加基本系统模块

step 2 [illegible]

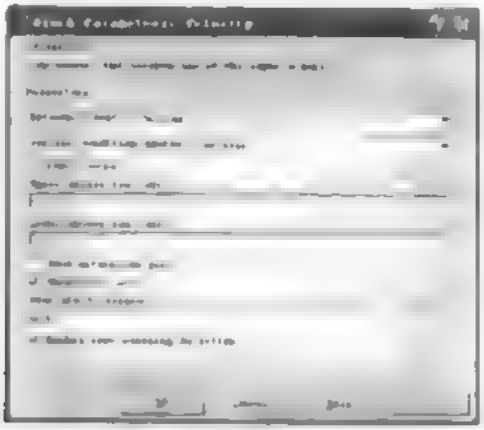


图 8.154 设置第一个积分器模块的属性

[illegible]

[illegible]，需要[illegible]的条件[illegible]置，[illegible]其设置[illegible]条件端口。因此需要选中所有的相关端口。

Step 3 双击第二个积分器模块，在弹出的积分器参数对话框中，在其中设置属性，如图 8.155 所示。

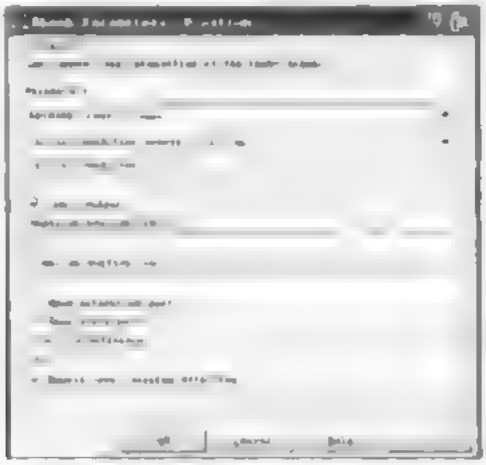


图 8.155 设置第二个积分器模块的属性

[illegible]

[illegible]

Step 4 设置[illegible]属性，因[illegible]的要求，需要[illegible]，所以需要将其坐标系的个数设置为 2，如图 8.156 所示。

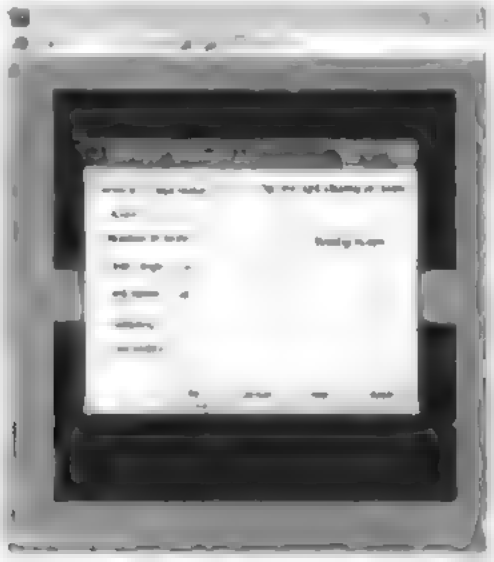


图 8.156 设置示波器的属性

之所以在上面的"Scope"中没有显示结果的原因，是因为在该步骤中还没有添加仿真工作。

step 5 查看仿真结果。将系统的仿真时间设置为20，然后选择模块窗口中的"Simulation"→"Start"命令，得到仿真结果，如图8.157所示。

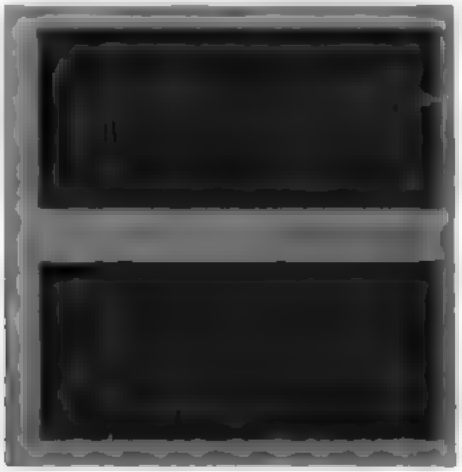


图8.157 查看仿真结果

在上面的仿真结果中，上面的图形表示小球运动速度随着时间变化的曲线，大致符合线性关系。下面的图形曲表示小球运动的高度随着时间变化的曲线，大致符合一次抛物线的关系。

step 6 简要分析该系统的原理。

本系统分析的是在初始高度为10米的地方以初速度15m/s向上抛掷一球的运动轨迹，根据基础物理知识，选择重力加速度为9.81m/s²。同时，考虑到空气阻力对小球运动的影响，每次进行积分的时域，将积分器的时间步"Time step"速度转换为前一个时间步的0.8倍，相当于小球会在每一次步骤中失去能量的损失，得到的结果就是有衰减的小球运动轨迹图形和速度图形。

在上面系统模块中，用到了"Initial Condition"模块来设置初始条件。该模块在连接时间，特别是积分器模块中会经常用到。关于该模块的使用方法，可以查看相应的帮助文件。

例8.19 创建Simulink系统，求解二阶微分方程 $x''(t)+0.4x'(t)+0.9x(t)=0.7u(t)$ 的方程解，其中 $u(t)$ 是脉冲信号，需要使用Simulink来求解函数 $x(t)$。

step 1 改写微分方程。将上面需要求解的微分方程改为下面的形式

$$-0.4x'(t)-0.9x(t)+0.7u(t)=x''(t)$$

step 2 使用Simulink来创建上面的微分方程，完成的系统模块如图8.158所示。

step 3 设置"Pulse Generator"模块的属性。双击上面系统模块中的"Pulse Generator"模块，打开相应的属性对话框，设置相应的模块属性，如图8.159所示。

在图8.159的对话框中，主要设置了"Pulse Generator"模块的信号幅度、关于脉冲信号的其他属性信息，请查看相应的帮助文件。

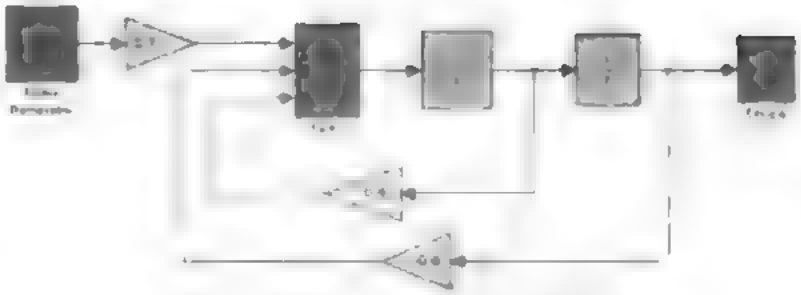


图 8.158　完成的系统模块

图 8.159　设置"Pulse Generator"模块的属性

Step 4 设置"Sum"模块的属性。双击"系统模块"中的"Sum"模块，打开其属性对话框，设置相应的模块属性，如图 8.160 所示。

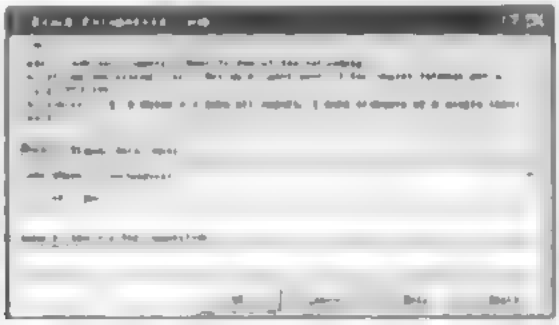


图 8.160　设置"Sum"模块的属性

在全部模块属性设置完成后，将模块的"Icon shape"属性设置为"rectangular"，这样模块显示效果，然后在"Icon text"文本框中输入"Sum"。

Step 5 查看仿真结果。将系统模型仿真时间设置为 1，然后选择模块界面工具栏的"Simulation"→"Start"命令，得到仿真结果，如图 8.161 所示。

Step 6 添加新的模块，将 8.161 仿真结果传输到"MATLAB 工作空间"。在上面仿真系统模块的基础上，添加"Clock"和"To Workspace"模块，将仿真结果传输到工作空间中，如图 8.162 所示。

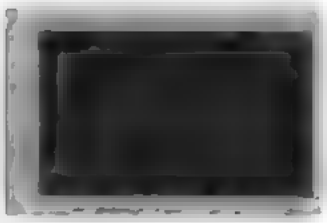


图 8.161 仿真结果

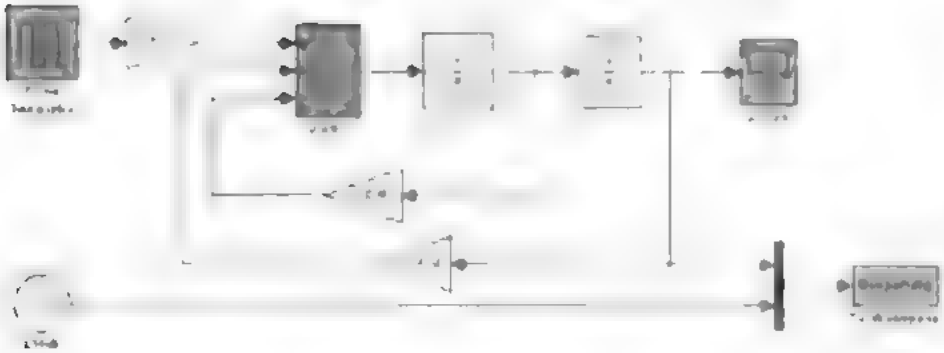


图 8.162 添加新的模块

[illegible]模块的[illegible]，在模块[illegible]的[illegible]，通过"To workspace"模块传送给了[illegible]变量 ScopeData。

step 7 设置"To workspace"模块的属性 [illegible]"To workspace"模块，打开模块的属性对话框，在其中设置生成的属性，[illegible]

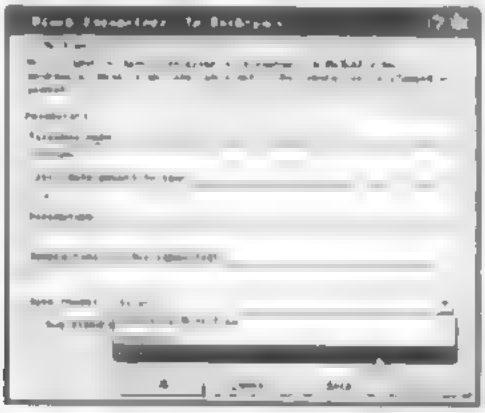


图 8.163 设置"To Workspace"模块的属性

在[illegible]中，将仿真结果的输出变量名称设置为"ScopeData"，[illegible]的格式设置为"Array"，用来在仿真结束后的[illegible]输出数值结果。

step 8 单击输出数据，自生成一个新的仿真系统，然后[illegible]，输入下面的程序代码。

```
>> clf;
>> t=ScopeData(:,1);
>> x=ScopeData(:,2);
>> [xm,km]=max(x);
>> plot(t,x,'m','LineWidth',3),hold on
>> plot(t(km),xm,'y.','MarkerSize',24),hold off
```

```
>> grid
```

step 9 [illegible]

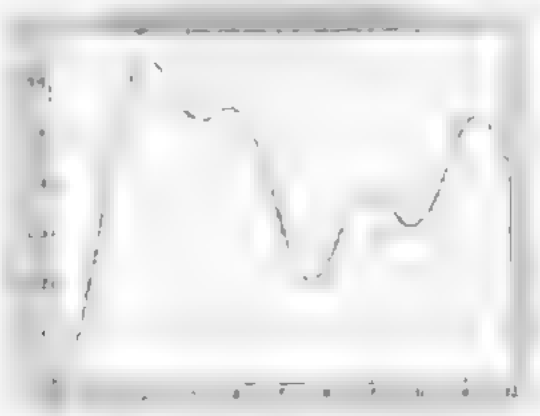


图 8.164　命令窗口中的绘制结果

[illegible]

- [illegible]
- [illegible]
- [illegible]

[illegible]

step 10 [illegible]

图 8.165　设置数据输出的选项

step 11 [illegible]

```
[illegible]
ScopeData =
[illegible]
```

```
      signals: [1x1 struct]
    [illegible]: '[illegible]'
>> ScopeData.signals
ans =
         values: [61x2 double]
     dimensions: 2
          label: ''
>> ScopeData.signals.values
ans =
          0            0
     0.0000       0.0002
     0.0000       0.0012
     0.0000       0.0062
     0.0003       0.0313
     0.0084       0.1564
     0.0421       0.3569
     0.0986       0.5569
     0.1742       0.7569
     [illegible]  [illegible]
     [illegible]  [illegible]
     [illegible]  [illegible]
     [illegible]  [illegible]
     [illegible]  [illegible]
     ...  ..........  ........  //限于篇幅，这里省略了部分数据
     [illegible]  [illegible]
     [illegible]  [illegible]
     [illegible]  [illegible]
     [illegible]  [illegible]
     0.5149       [illegible]
     0.4850       [illegible]
     0.4417       [illegible]
     [illegible]  [illegible]
```

提示

从上面的结果可以看出，如果选择 [illegible] 上的 'Time' 和 '[illegible]' 选项，[illegible] 通过对象的属性直接调用仿真数据，[illegible] ScopeData.time [illegible] 仿真的时间和仿真数据

例 8.20 使用传递函数模块，求解例 8.19 中的实例。

step 1 [illegible]

$x''(t)+0.4x'(t)+0.9x(t)=0.7u(t)$

将上面的微分方程的两边同时进行变换，得到的结果如下

$s^2X(s)+0.4sX(s)+0.9X(s)=0.7U(s)$

将上面的方程进行整理，得到转换公式如下

$$G(s)=\frac{X(s)}{U(s)}=\frac{0.7}{s^2+0.4s+0.9}$$

step 2 根据上面的转换函数公式，[illegible] 系统的模块，[illegible]

图 8.166 创建系统的模块

step 3 [illegible] 系数的公式，如图 8.167 所示。

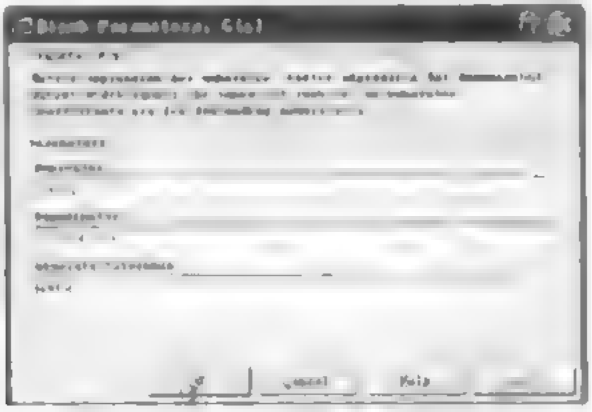


图 8.167　设置转移函数的属性

step 4 [illegible] 如图 8.168 所示。

step 5 [illegible] 如图 8.169 所示。

图 8.168　设置属性后的模块

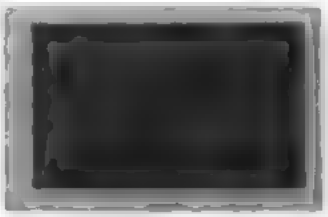


图 8.169　查看仿真结果

[illegible]

例 8.21　使用状态方程模块来求解例 8.19 中的实例。

step 1 [illegible]

令 $x(1)=x$，$x(2)=x'$，则 $x''(t)+0.4x'(t)+0.9x(t)=0.7u(t)$ 转换为

$$\dot{x}=\begin{bmatrix}\dot{x}(1)\\ \dot{x}(2)\end{bmatrix}=\begin{bmatrix}x(2)\\ -0.9x(1)-0.4x(2)+0.7u(t)\end{bmatrix}=\begin{bmatrix}0 & 1\\ -0.9 & -0.4\end{bmatrix}\begin{bmatrix}x(1)\\ x(2)\end{bmatrix}+\begin{bmatrix}0\\ 0.7\end{bmatrix}u(t)$$

将上面的方程转换为下面的状态方程

$$\begin{cases}\dot{x}=Ax+Bu\\ y=Cx+Du\end{cases}$$

[illegible]，系数 $A=\begin{bmatrix}0 & 1\\ -0.9 & -0.4\end{bmatrix}$，$B=\begin{bmatrix}0\\ 0.7\end{bmatrix}$，$C=\begin{bmatrix}1\\ 0\end{bmatrix}$，$D=0$。

step 2 根据上面的状态方程，[illegible]个线性系统子模块，如图所示。

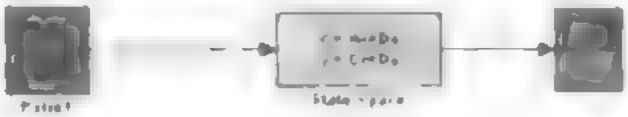


图 8.170 添加系统的模块

在 Simulink 中，任何一个模块都是在[illegible]模块下的子模块，它表示一个的一个线性状态系统。

step 3 设置状态方程模块的属性，[illegible]，在其中设置转换与数的公式，如图 8.171 所示。

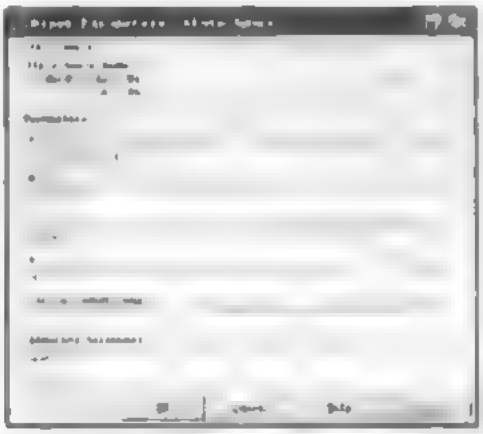


图 8.171 设置状态方程模块的属性

step 4 查看仿真结果。[illegible]模块，查看仿真结果，[illegible]所示。

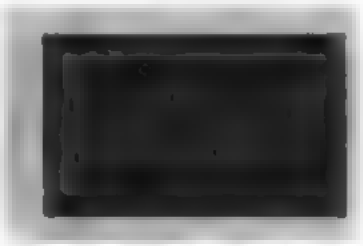


图 8.172 查看仿真结果

[illegible]Simulink 中，[illegible]可以灵活地使用各种模块完成建模工作。

8.6.3 "积分器"模块的工作原理

根据上一小节的介绍，在连续系统建模中，"Integrator"模块是一个十分重要的模块，[illegible]用"Integrator"模块可以完成各种复杂的连续系统。[illegible]"Integrator"模块各种属性对系统建模

块是个十分重要的内容，在本小节中将详细介绍"Integrator"模块的各种常见属性参数设置工作。

在Simulink中，"Integrator"模块的功能就是对信号进行积分，选择"Continuous"模块库的"Integrator"模块，然后添加至模型编辑界面中，得到的默认模块如图8.173所示。

图8.173 默认的"Integrator"模块

"Integrator"模块输出其输入信号在当前时间步（time step）内的积分结果，可以使用一个简单的公式来描述其工作原理

$$y(t)=\int_{t_0}^{t}u(t)dt+y_0$$

在上面的公式中，$y(t)$表示的是"Integrator"模块的输出信号，也就是积分结果，$u(t)$表示"Integrator"模块的输入信号，y_0则表示积分的初始状态，其中，$y(t)$和$u(t)$都是关于时间t的函数表达式。

在Simulink中，可以使用多种数值积分方法来计算"Integrator"模块的输出信号，每种方法各有优劣。在前面介绍过，在"Configuration Parameters"对话框的Solver面板中，可以选择不同的数值积分方法。Simulink将"Integrator"模块作为某个状态方程的动态系统，其输入信号是该系统状态的时间导数值。下面的方程组可以表示该原理

$$x=y(t)$$

$$x_0=y_0$$

$$x'=u(t)$$

除了"Integrator"模块的默认属性设置之外，"Integrator"模块还提供了多种初始状态的设置模块，默认的初始状态值为0，使用"Integrator"模块的模块属性对话框，用户可以根据实际需要的需要设置新的初始状态数值，或者为"Integrator"模块添加初始状态数值端口。在"Integrator"的模块属性对话框中，一般可以设置

- 设置积分的上限和下限
- 添加一个输入端口来重置积分的初始状态数值
- 创建一个可选的状态输出端口，状态输出端口一般用来为发生事件时的重置
-

8.6.4 设置初始状态数值

除了上面的设置工作之外，在模块的属性对话框中，还可以设置关于积分的其他各种属性，下面将介绍设置积分的初始状态数值。

用户可以在模块对话框中设置积分的初始状态数值，或者引入外部的信号作为初始状态。如果将积分的初始状态作为模块的参数，则可以在对话框"Initial condition source"下拉菜单中选择"internal"，然后在"Initial condition"选项中输入相应的初始数值。如果从外部信号中引入初始的初始数值，则可以在"Initial condition source"下拉菜单中选择"external"，如图8.174所示。

在完成上述的属性设置以后，单击"Apply"按钮，积分器模块自动添加一个初始状态的端口，如图8.175所示。

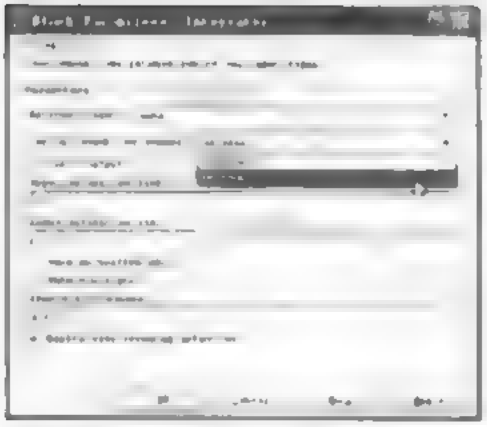


图 8.174 设置外部信号的初始状态

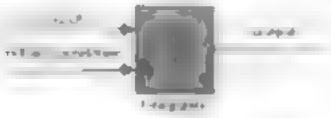


图 8.175 添加初始状态端口

8.6.5 设置积分限制

[illegible]

根据设置的不可积分限制，分下面几种情况来分析：

- [illegible]被保持为下限数值。
- [illegible]
- [illegible]数值。

[illegible] port" 选项，如图 8.176 所示。

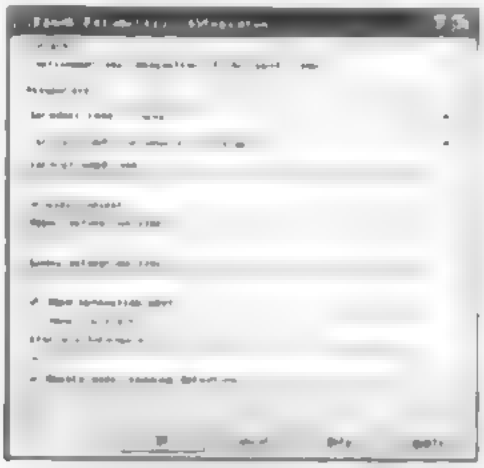


图 8.176 设置显示信号端口

[illegible]单击对话框中的"Apply"按钮后，模块的模块图标如图 [illegible]

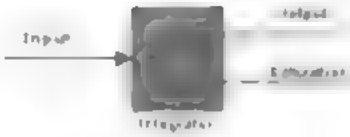


图 8.177　添加信号端口

在上面步骤中添加的"Saturation"端口可以输出三种信号结果：

- 1：表示在积分器中设置了上限
- 0：表示积分器没有设置任何的积分限制
- -1：表示在积分器中已经设置了下限。

[illegible]

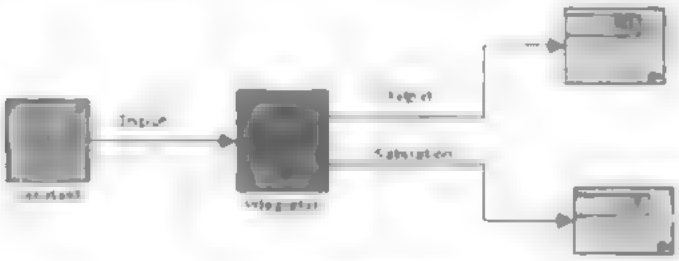


图 8.178　添加简单的系统模块

[illegible]

8.6.6 重设积分状态

[illegible]

[illegible]

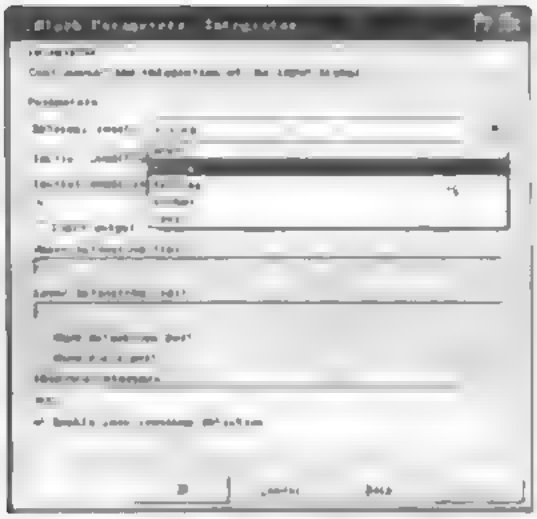


图 8.179 设置积分器的重设属性

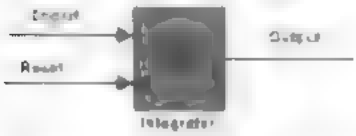


图 8.180 设置后的积分模块

8.6.7 设置积分状态端口

为了各种积分功能，Simulink允许用户设置并显示积分器模块的积分状态端口，积分器模块的积分状态端口[illegible]

可以在模块对话框中选择"Show state port"选项，来显示积分器模块的积分状态端口，如图8.181所示。

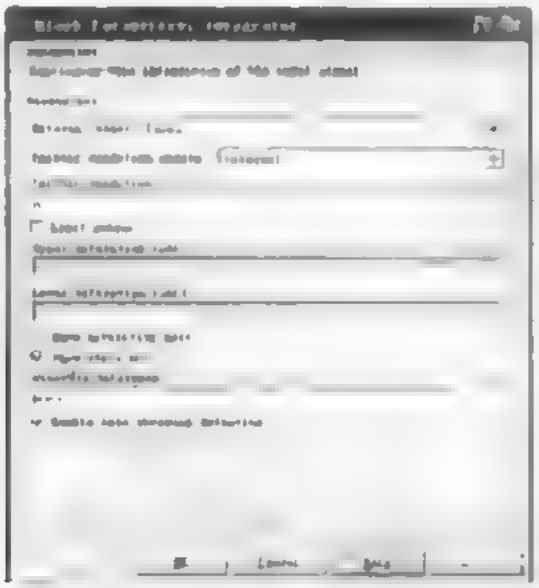


图 8.181 设置积分器的积分状态端口

[illegible]

在 Simulink 中，一般有两种常见的情况需要使用到状态端口：

- 自我循环系统（见步骤 8.17）。
- [illegible]

[illegible]

[illegible]

[illegible]

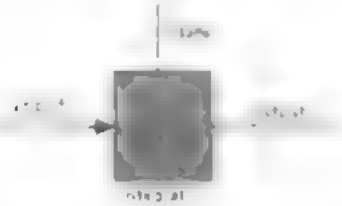


图 8.143 [illegible]

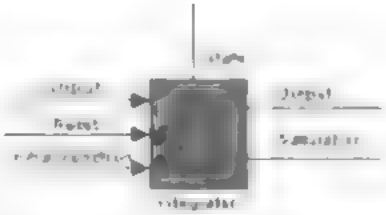


图 8.143 [illegible]

[illegible]

8.7 非线性系统建模

[illegible]

8.7.1 非线性系统建模简介

例 8.22 [illegible]

step 1 [illegible]

$$\frac{1}{4}(3x-2x^3)x'-x=x''$$

step 2 [illegible]

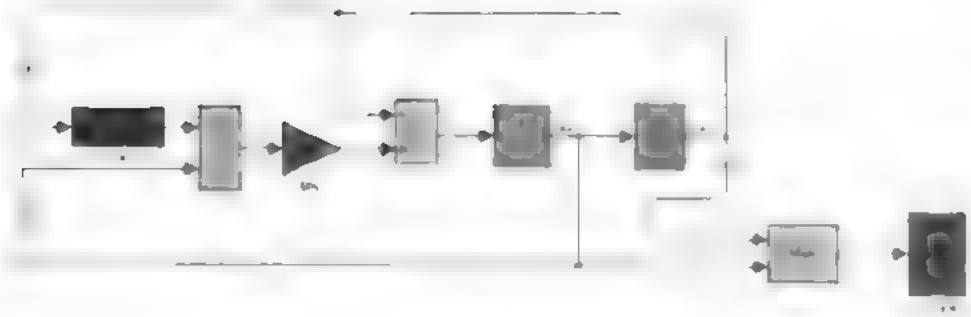


图 8.184 创建的系统模块

[illegible]

[illegible]

Step 3 设置"Fcn"模块的属性。单击"图系统"中的"Fcn"模块，打开其属性对话框，在其中设置表达式，如图 8.185 所示。

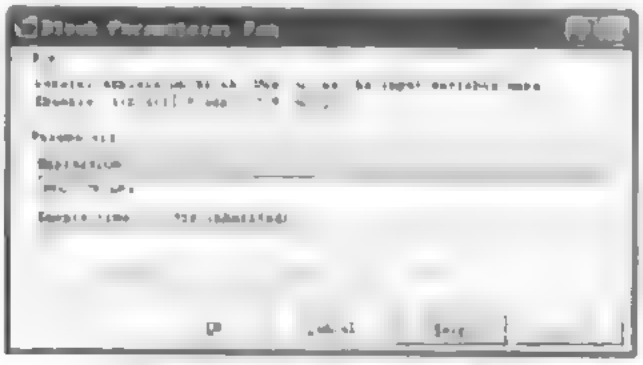


图 8.185 设置函数表达式

[illegible]

[illegible]

Step 4 设置"Product"模块的属性。[illegible]，在其中设置表达式，如图 8.186 所示。

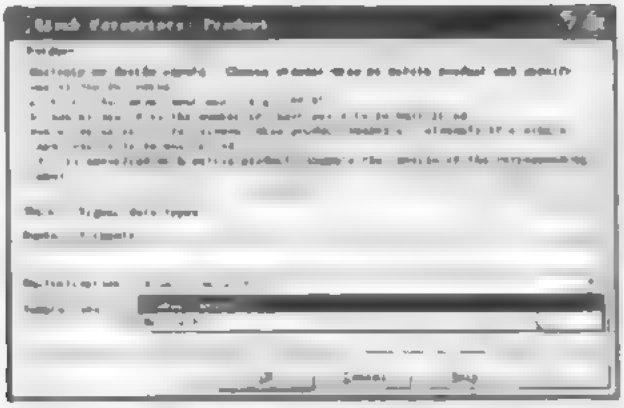


图 8.186　设置乘法模块的属性

在[illegible]中，选择"Product"模块，[illegible]属性为"[illegible]"，[illegible]模块输入端口数目为2，在"[illegible]"选项中输入[illegible]，选择"Product"模块会提供2个输入端口。

[illegible]

Step 5　设置"[illegible]"模块的属性，在[illegible]模块使用[illegible]属性，[illegible]要求设置初始数值条件，以模块"x"为例，双击该模块，[illegible]属性对话框，设置的初始数值如表 8.187 所示。

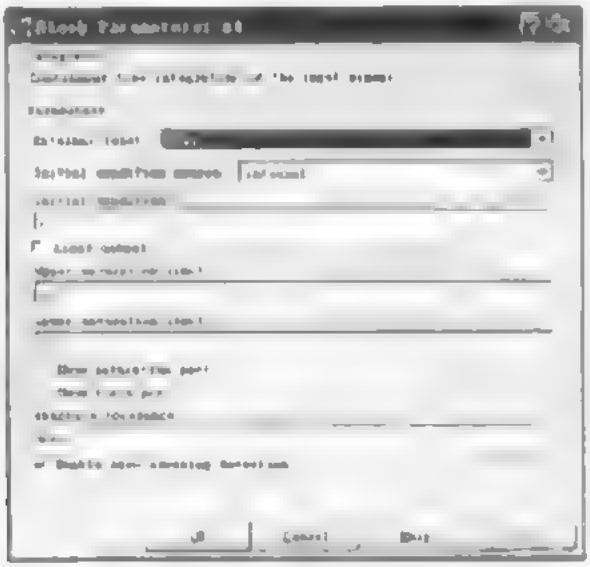


图 8.187　设置积分器模块的属性

[illegible]

Step 6　[illegible]，选择"Scope"模块，查看仿真结果，如表 8.188 所示。

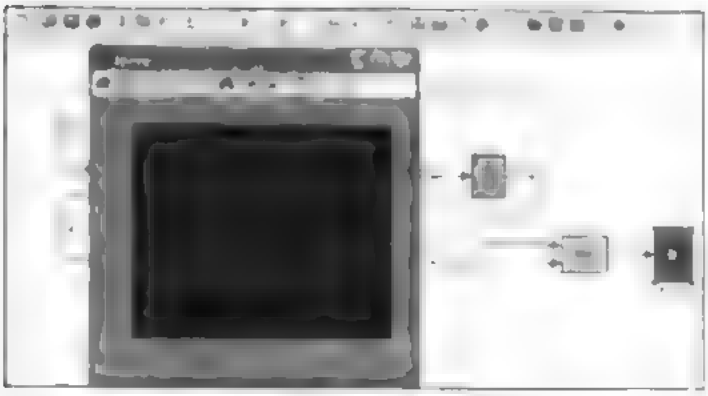


图 8.188 查看仿真结果

在上面的仿真结果中，黄色的曲线表示变量 $x(t)$，红色的曲线表示 $\dot{x}(t)$。

step 7 修改仿真的模块。为了能够在 MATLAB 的工作空间中查看[illegible]，[illegible]模块，如图 8.189 所示。

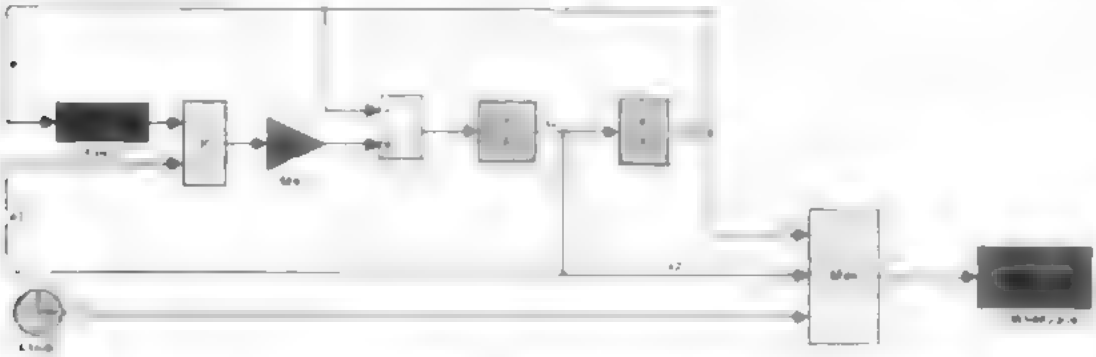


图 8.189 添加新的模块

step 8 设置"Mux"模块的属性。为了在 MATLAB 工作空间中显示[illegible]，将"Mux"模块的输入端口改为 3，如图 8.190 所示。

图 8.190 修改"Mux"模块的输入端口

step 9 将系统仿真时间修改为 40，运行系统仿真，将输出变量"[illegible]"，然后在 MATLAB 的命令窗口中输入下面的代码：

```
>> x=[illegible]data(:,[illegible]);
>> dx=[illegible]data(:,[illegible]);
>> [illegible]data(:,[illegible]);
>> plot(t,x,'[illegible]',t,dx,'y',[illegible],...,[illegible])
>> grid
>> xlabel('时[illegible]')
>> legend('x','dx'
```

Step 10 [illegible]

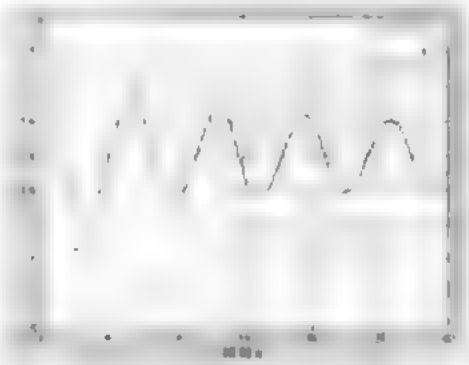


图 8.191 绘制图形结果

[illegible]

8.7.2 非线性系统建模实例

[illegible]

例 8.23 [illegible]，而且该物体的摩擦力微分方程如下

$$\frac{1}{\delta}(kx-F\_fr)=x''$$

[illegible]

[illegible]

Step 1 [illegible]

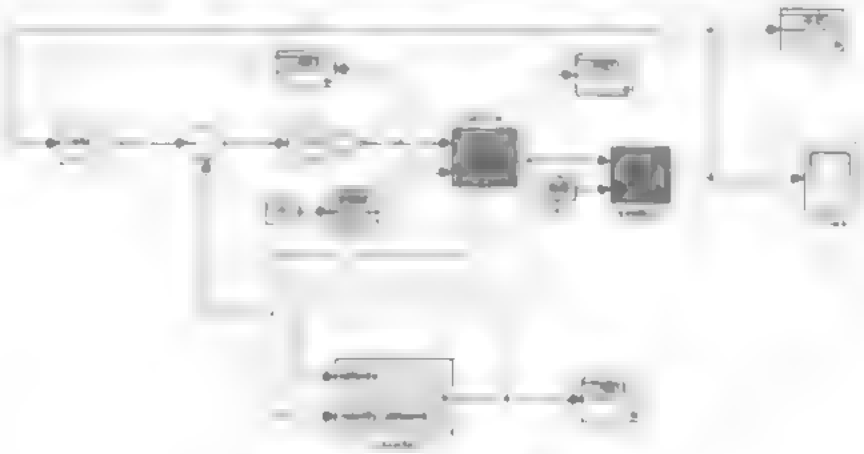


图 8.192 添加微分方程的模块

在上面的系统模块中，[illegible]，其中第一条[illegible]系统模块的表达式为[illegible]，[illegible]$\frac{1}{s}(kx-k\_fr)=x$，[illegible]物体位移到达[illegible]，在本步骤中，[illegible]，公式中 $k\_fr=k\_fr(x)$，[illegible]计算 k_fr 的数值，本例[illegible]子系统模块，[illegible]子系统模块中的输入变量名称 velocity 和 velocity_to_import，其中变量 velocity，velocity_to 是在"建立仿真模型"步骤中已经定义的变量，而在"velocity"模块的"状态"选项卡[illegible]输入变量。

step 2 双击"velocity"模块的[illegible]，双击"velocity"模块，打开模块的属性对话框，[illegible]其设置"状态"选项卡，如图 8.193 所示。

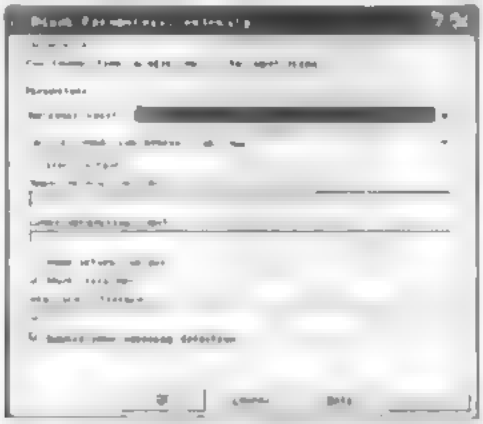


图 8.193 设置积分器模块的属性

在上面的系统模块中，[illegible]"Display"模块，[illegible]模块[illegible]模块[illegible]，但是，[illegible]量的数值，[illegible]仿真系统。

step 3 设置计算摩擦力的子系统模块，[illegible]子系统模块，在[illegible]模块编辑器中添加子系统模块，如图 8.194 所示。

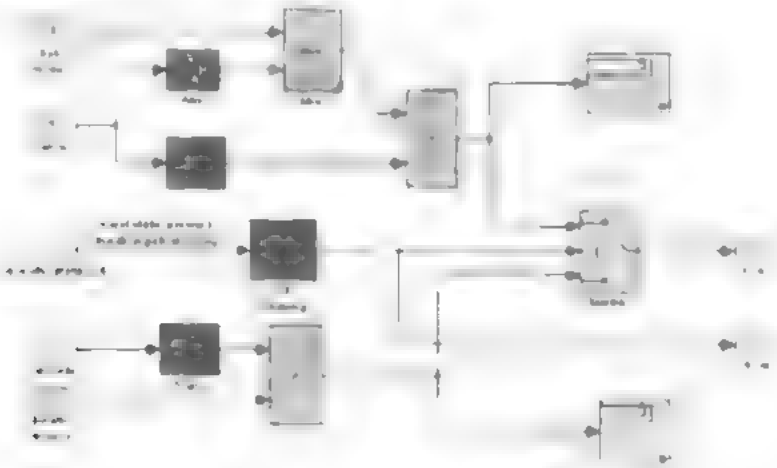


图 8.194 添加计算摩擦力的子系统模块

step 4 参数的[illegible]模块的参数，[illegible]系模块[illegible]，[illegible]系模块名称[illegible]，[illegible]，[illegible]如图 8.195 所示。

图 8.195　添加子系统模块后的主系统模块

step 5　[illegible]

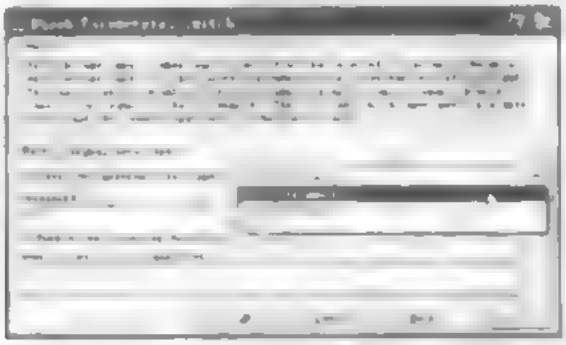


图 8.196　设置"Switch"模块的功能

step 6　[illegible]

[illegible]

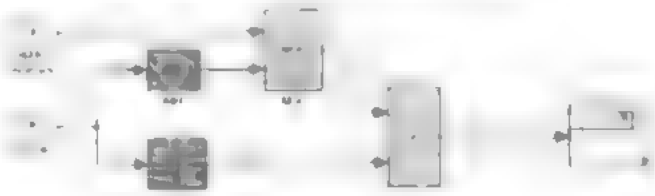


图 8.197 子系统的上部信号系统

[illegible]

Step 7 [illegible]

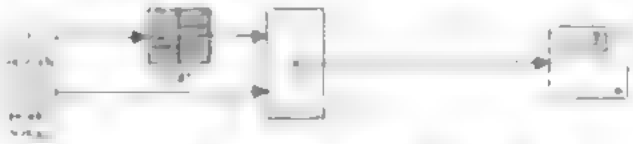


图 8.198 子系统的下部信号系统

[illegible]

Step 8 [illegible]的模块系统如图 8.199 所示。

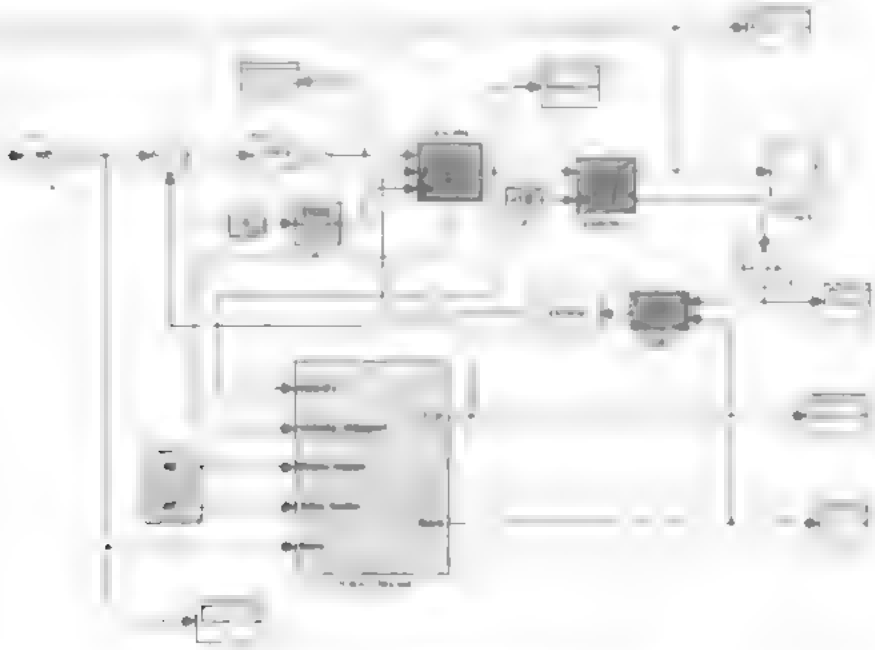


图 8.199 设计完成的系统模块

step 9 设置 "Velocity" 积分器模块的属性。双击 "Velocity" 积分器模块，打开对应的属性对话框，如图 8.200 所示。

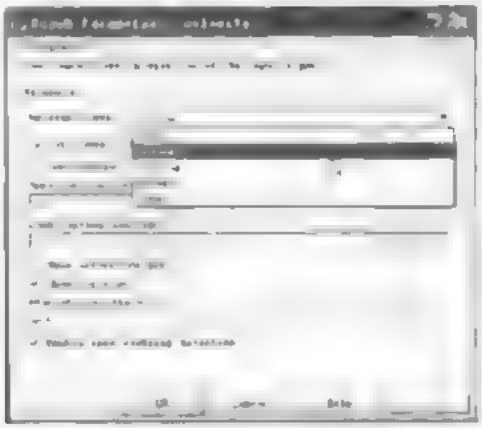


图 8.200 设置 "Velocity" 积分器模块的属性

[illegible]，[illegible]详细设置对应的逻辑模块实现。

step 10 设置 "Position" 积分器模块的属性。双击 "Position" 积分器模块，打开对应的属性对话框，在其中设置其属性，如图 8.201 所示。

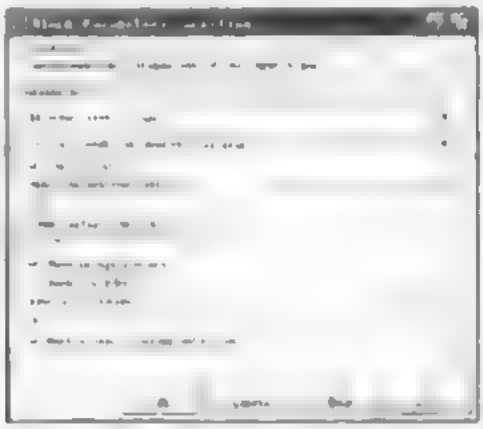


图 8.201 设置 "Position" 积分器模块的属性

选中 "Show saturation port" 选项，积分器模块出现 "Saturation Port"，[illegible]该端口输出的数据类型是布尔数据类型。

step 11 设置 "Data Type Conversion" 模块的属性。[illegible] "Data Type Conversion" 模块，[illegible] "boolean" [illegible] "double" [illegible]。[illegible] "Output data type mode" 选择 "double" 数据类型，[illegible]。[illegible] "Output data type mode" 设置为 "boolean" 数据类型即可。

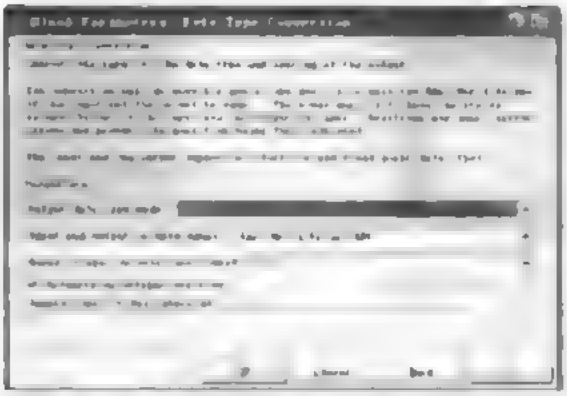


图 8.202 设置数据转换模块的属性

"Data Type Conversion" 模块在 Simulink 模型中发挥[illegible]数据类型之间的转换，[illegible]模块可以实现多种功能，[illegible]

step 12 设置仿真时间，运行整个仿真系统，[illegible]仿真，得到的结果如图 8.203 所示。

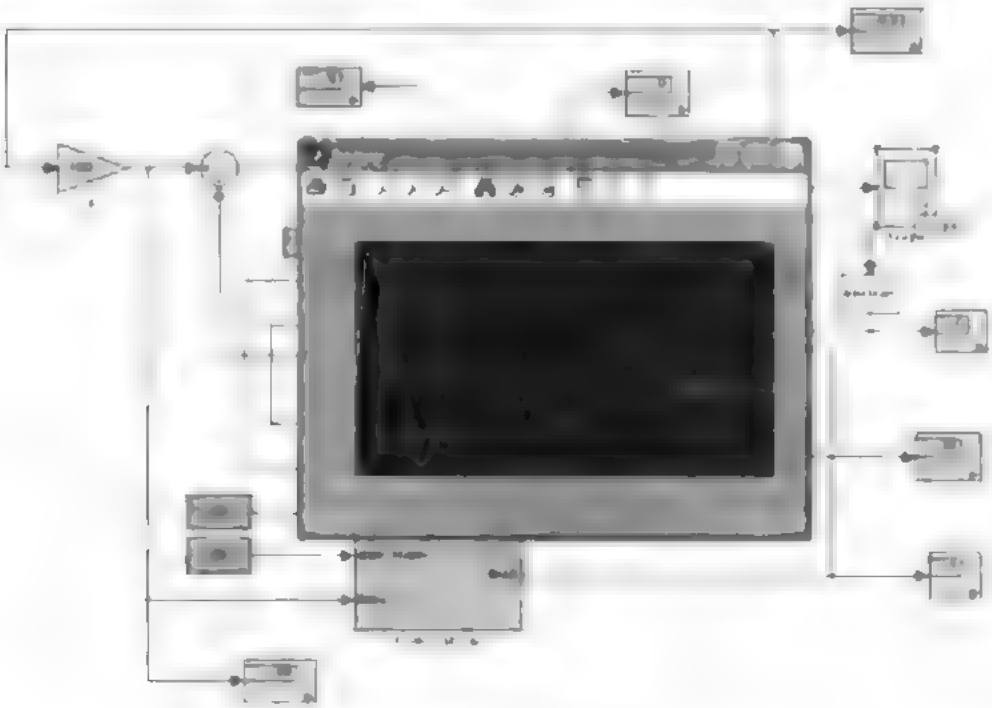


图 8.203 查看仿真结果

在上面的仿真结果中，各个[illegible]模块的[illegible]，[illegible]的[illegible]、位移、速度[illegible]各个物理参量的数值。[illegible]模块[illegible]

step 13 修改坐标轴数值范围，在"[illegible]"模块[illegible]的情况，[illegible]修改坐标轴数值的显示范围，如图 8.204 所示。

step 14 查看修改后的波形，单击[illegible]"Apply"按钮，查看结果，[illegible]

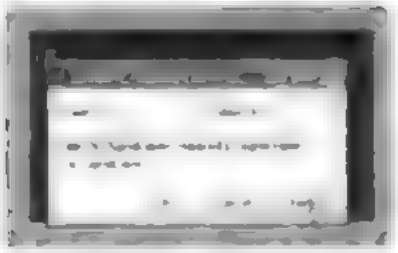


图8.204　修改图形的Y轴坐标数值

图8.205　修改后的最大图形

[illegible]

Step 15 修改系统仿真参数，保存新的仿真结果，观察[illegible]

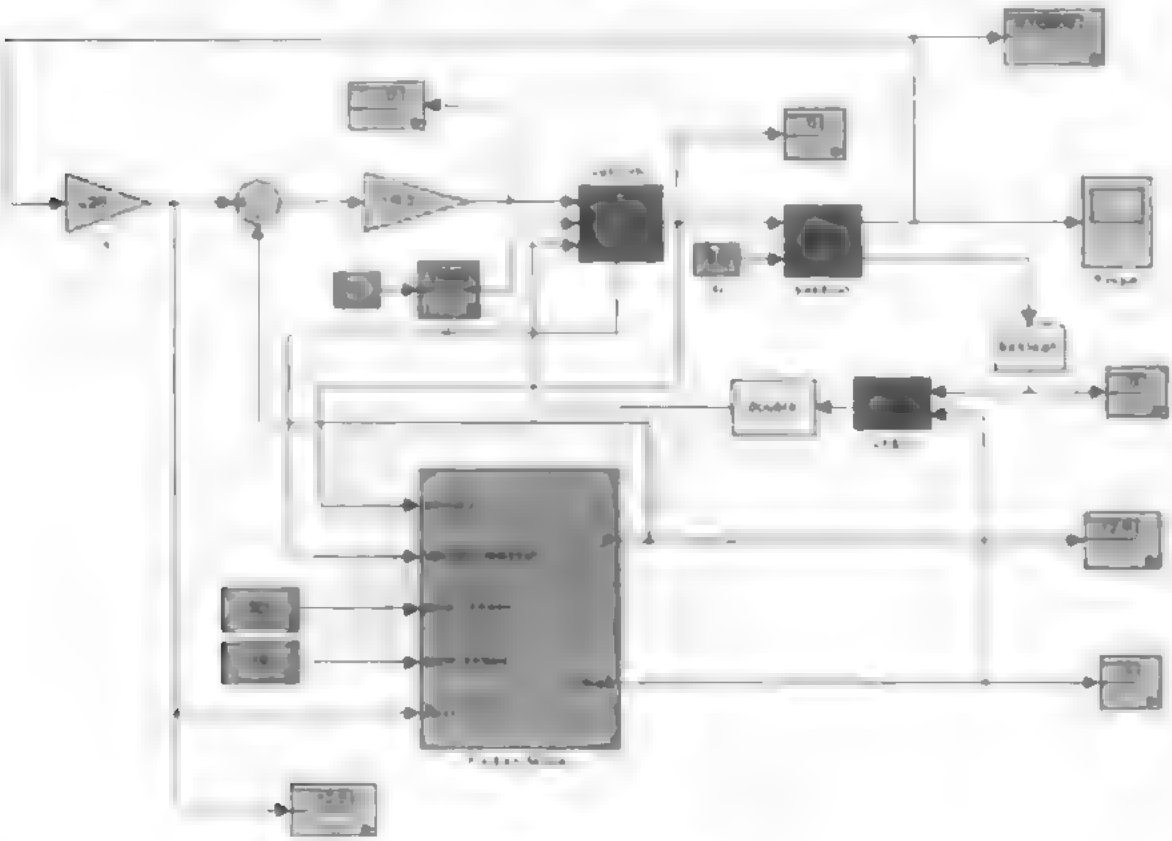


图8.206　修改仿真参数

[illegible]

Step 16 [illegible]

[illegible]

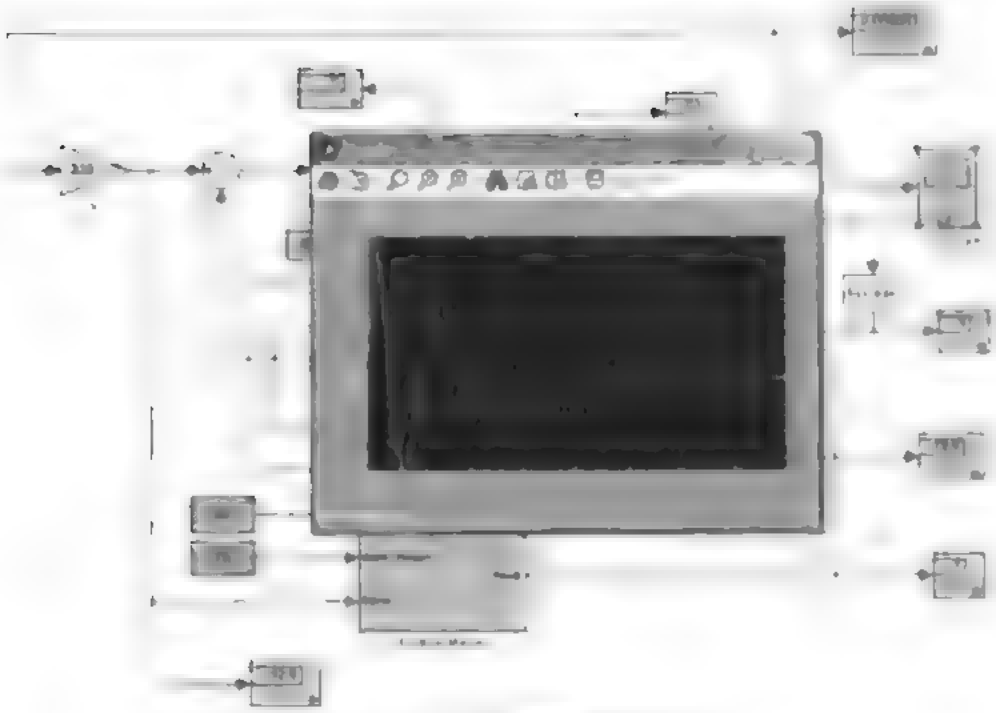


图 8.207 雷达运行系统的仿真

8.8 小结

[illegible]

◆ ◆ ◆

第9章　Simulink 仿真的高级技术

本章导读

- 子系统
- 使能子系统
- S函数
- 封装子系统
- 触发子系统
- 仿真结果分析

[illegible]

9.1 子系统

[illegible]

9.1.1 子系统的基础知识

[illegible]

- 减少模块窗口中模块的个数，使得模型窗口更加简洁
- [illegible]
- 增强整个系统运行的效率和可靠性
- [illegible]

在仿真模型中的子系统的作用，就如同在MATLAB中编写程序的函数，[illegible] function [illegible] -prog [illegible]，用户在MATLAB命令窗口中输入 [illegible] 等

在Simulink中，可以使用如下两种方法创建子系统。

- [illegible]

块并向其中添加模块。

- 结合已经在模块编辑器中添加的模块，创建子系统。

在Simulink中，[illegible]子系统的方式，[illegible]范围，这样当获得直观的效果[illegible]

9.1.2 使用子系统模块创建子系统

例9.1 使用Simulink中的Subsystem模块来一建系统，系统实现输入变量n为，输出变量y从1到N的自然数的累积求和数值，下面分步骤详细介绍。

step 1 单击"Simulink Library Browser"窗口工具栏中的"File"→"New"→"Model"命令，打开一个空白模型窗口，[illegible]系统模块，如图9.1所示。

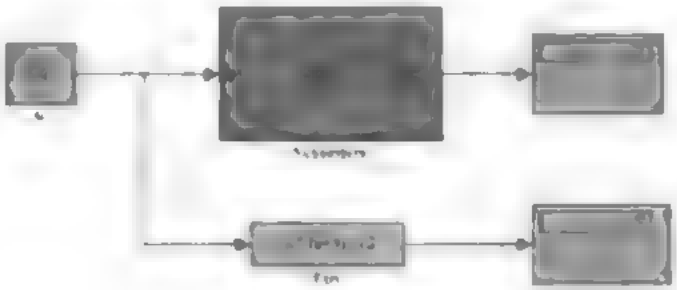


图9.1 添加程序模块

在上面模块中的"Subsystem"子模块属于"Ports & Subsystems"模块库的"Interface Subsystem"模块，该模块主要功能是[illegible]。本例选用的是该模块的默认属性，如图9.2所示。

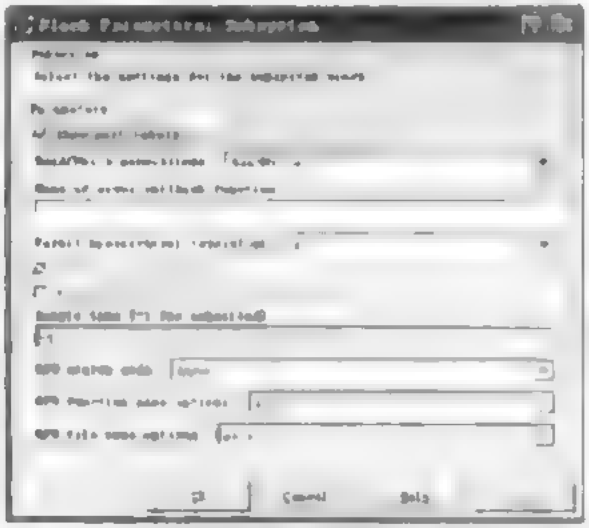


图9.2 Subsystem的属性对话框

step 2 设置"Fcn"模块的属性。在上面的程序模块中，为了检查子系统计算结果，在下面的分支中添加了"Fcn"模块，属于"User-Defined Functions"模块库下的"Fcn"模块，其模块属性如图9.3所示。

图9.3 "Fcn"模块的属性

根据基础的数学知识，自然数从1到N的叠加求和公式为

$$1+2+3+\quad+N=\frac{N(N+1)}{2}$$

对比，在上面的"Fcn"模块中，如果要生成公式所要的求和数值结果。

Step 3 [illegible]

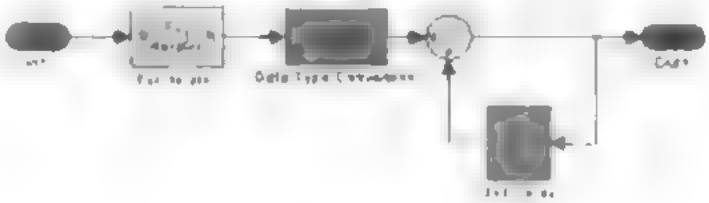


图9.4 编写子系统模块

Step 4 设置"For Iterator"模块的属性。对于此处，在创建子系统模块时，选择"For Iterator"模块，选择该模块的部分，[illegible]，选择以下属性对话框中的[illegible]

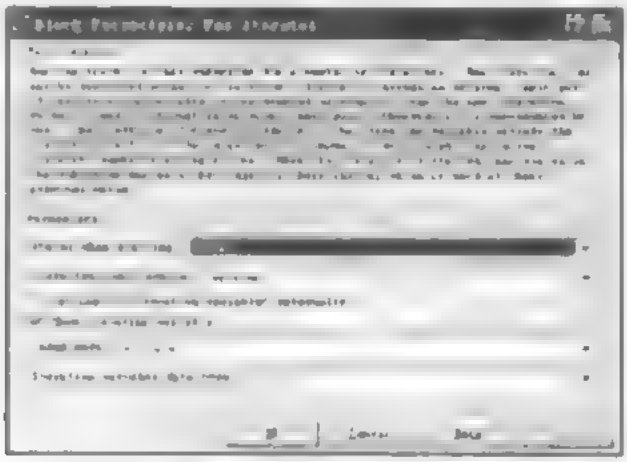


图9.5 设置"For Iterator"模块的属性

[illegible]

[illegible]

Step 5 [illegible]

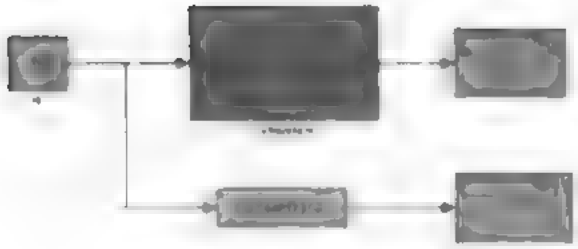


图 9.6 查看系统仿真的结果

当输入 [illegible] 自然数的数值是 [illegible]，仿真系统得出结果 55。[illegible] 输入 [illegible] 自然数的数值，如 20，重新运行仿真，得到的结果如图 9.7 所示。

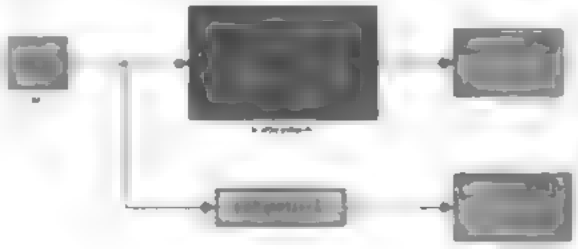


图 9.7 重新运行仿真

以上，在结果中可以看出，当输入 [illegible] 自然数 [illegible]。[illegible] 在这个示范的简单，但是表示了创建子系统的一般步骤，具体的内容为

step 1 从模块库中选择合适的子系统模块，添加到模块编辑器中

step 2 双击相应的子系统模块，打开子系统的模块编辑器

step 3 在模块编辑器中添加模块，创建子系统，然后保存

step 4 运行仿真，得到结果。

[illegible] 子系统模块，[illegible] 模块 [illegible] 子系统模块 [illegible] 这些内容将在后面章节给出。

9.1.3 使用模块组合子系统

例 9.2 使用 Subsystem 编辑子系统模块，并编辑 [illegible] 序列 $\sin x + \frac{1}{2}$ [illegible] 的数值。

step 1 单击 "Simulink Library Browser" 工具栏中的图标，在弹出的编辑器中选择 "File→New→Model" 命令，打开一个空白模型窗口，然后添加子系统模块，[illegible]

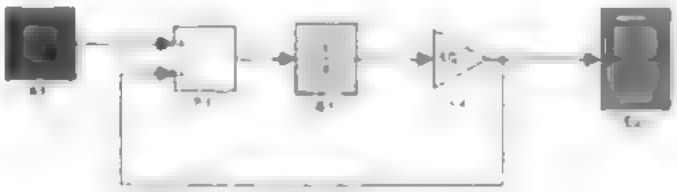


图 9.8 添加相应的系统模块

[illegible]

Step 2 运行上面的仿真系统，得到的结果如图9.9所示。

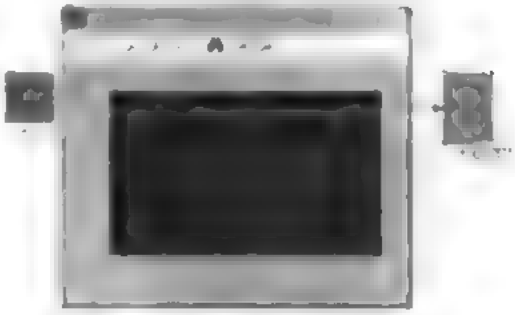


图9.9 查看系统的仿真结果

Step 3 [illegible]子系统，如图9.10所示。

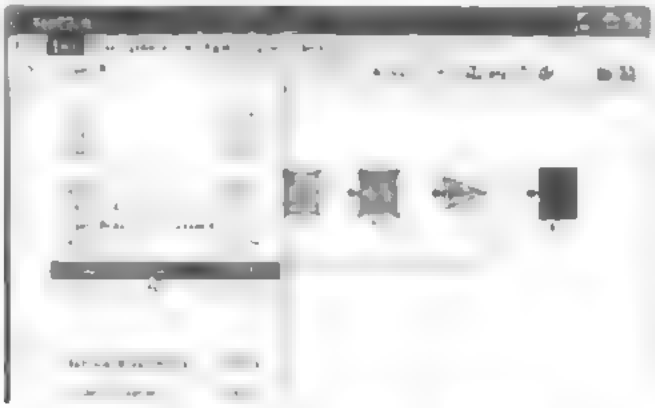


图9.10 创建子系统模块

Step 4 [illegible]，如图9.11所示。

图9.11 创建子系统

Step 5 [illegible]动创建的子系统模块，如图9.12所示。

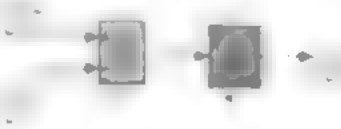


图9.12 Simulink创建的子系统

[illegible]

step 6 [illegible]，得到的仿真结果如图 9.13 所示。

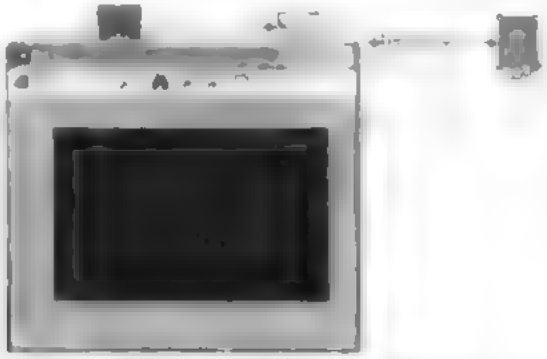


图 9.13 重新运行系统的仿真

[illegible]

step 1 [illegible]

step 2 [illegible]

step 3 双击创建的子系统模块，编辑子系统的名称或者其他属性

step 4 运行仿真，得出结果。

[illegible]

9.2 子系统实例

[illegible]

9.2.1 添加控制信号

例 9.3 [illegible]

step 1 [illegible]

step 2 [illegible]中设置方形波的参数属性，如图 9.15 所示。

[illegible]

[illegible]在 Simulink 中，"Pulse Generator"模块[illegible]主要参数[illegible]Amplitude（幅值）、Pulse Width（脉冲宽度）、Period（周期）和 Phase Delay（相位延迟）等。参见图 9.16 了解到各属性的含义。

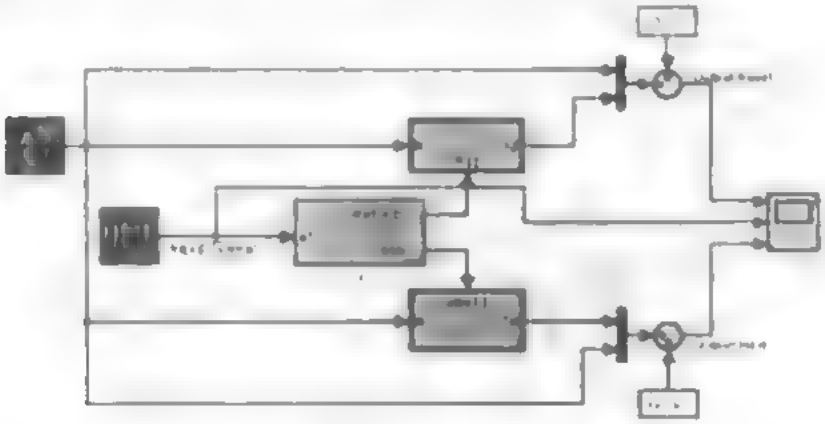


图 9.14　添加程序模块

图 9.15　方形波的属性

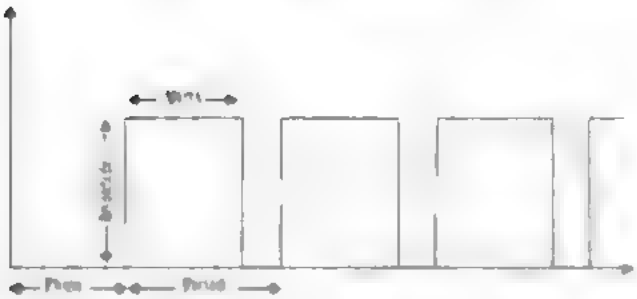


图 9.16　方形波的属性参数

step 3 设[illegible]在[illegible]参数的参数值，[illegible]参数数值，选择"Pulse Generator"模块，[illegible]，在弹出的快捷菜单中选择"Block Properties"选项，[illegible]"Pulse Generator"对话框，选择"Block Annotation"选项卡，在其不设置已[illegible]参数数值，[illegible]

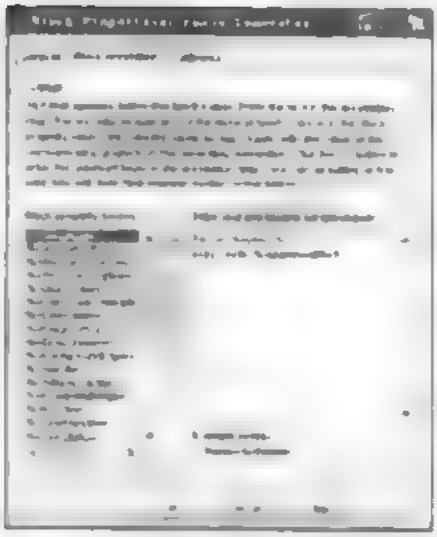


图 9.17 设置模块显示的参数数值

step 6 查看修改后的模块。单击上面对话框中的"Apply"按钮，[illegible]模块，如图 9.18 所示。

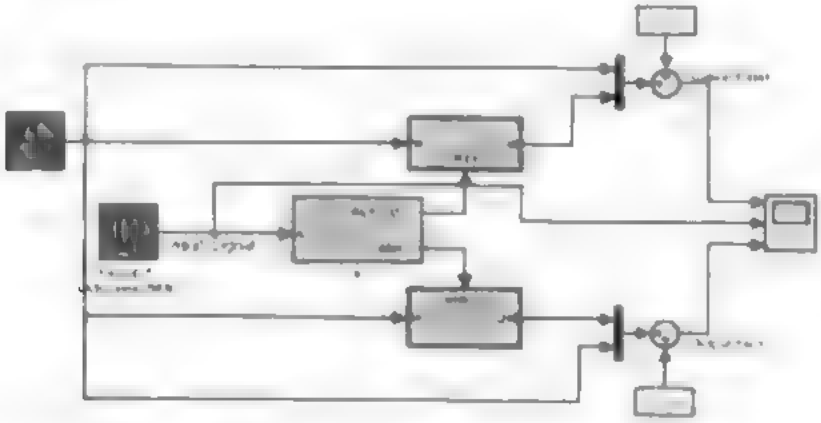


图 9.18 显示模块的主要参数

[illegible]

9.2.2 添加子系统模块

继续上面小节的步骤。

step 1 添加子系统模块的[illegible]，如图 9.19 所示。

step 2 设置"子"模块的属性。[illegible]模块，[illegible]设置该模块的属性。如图 9.20 所示。

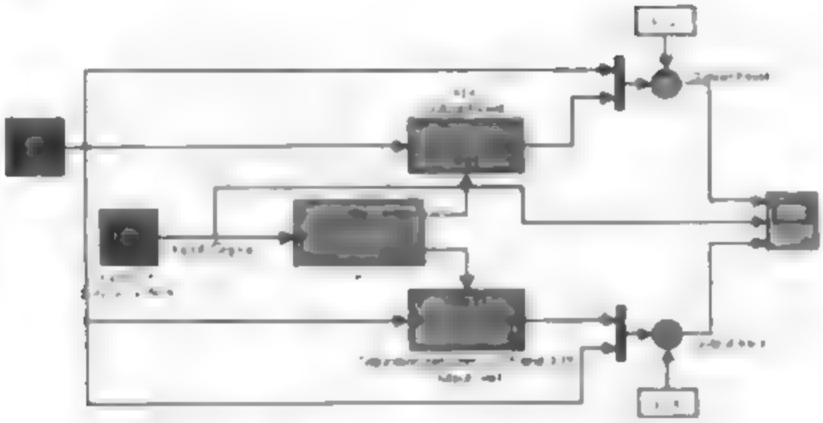


图9.19　添加子系统模块的说明文字

图9.20　设置"If"模块的属性

在[illegible]对话框中，将If模块的输入[illegible]设为"1"，[illegible]。因此，[illegible]输入[illegible]，[illegible]。有一个条件，[illegible]"Else condition"[illegible]。[illegible]，需要选择"Show else condition"选项，[illegible]。

step 3 分析If模块的功能。在Simulink中，"Ports & Subsystems"模块库下的If模块主要功能是在Simulink中执行类似于C语言中的if-else控制流程序语句。该模块和若干个包含执行[illegible]的"If Action"子系统模块结合[illegible]，实现if-else逻辑功能，其典型子模块结构系统如图9.21所示。

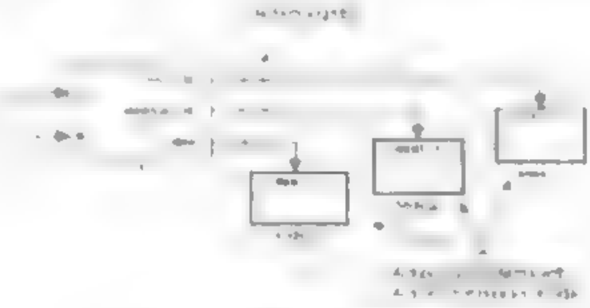


图9.21　If模块的典型执行结构

针对上面的系统模块结构，其功能和下面的程序代码相同。

```
if (u1 > 0)
    body_1;

elseif (u2 > 0)
    body_2;

else
    [illegible];
```

[illegible]

step 4 设置 "If Action" 子系统模块的属性。[illegible] 子系统模块，打开模块编辑器，添加相应的子系统模块，如图 9.22 所示。

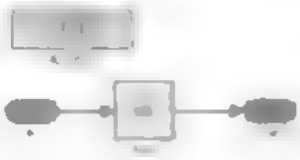


图 9.22 添加 "If Action" 子系统模块

step 5 设置 "Else Action" 子系统模块的属性。[illegible] 子系统模块，打开模块编辑器，添加相应的子系统模块，如图 9.23 所示。

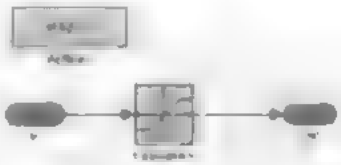


图 9.23 添加 "Else Action" 子系统模块

step 6 设置 Saturation 模块的属性。在上面的模块系统中，Saturation 模块主要作用是限定 Else 系统中的变量数值范围，其属性对话框如图 9.24 所示。

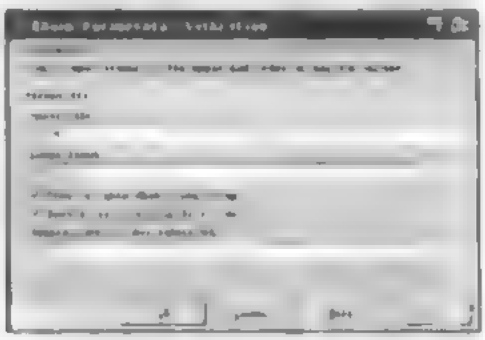


图 9.24 设置 Saturation 模块的属性

9.2.3 添加显示模块

[illegible]

Step 1 [illegible]

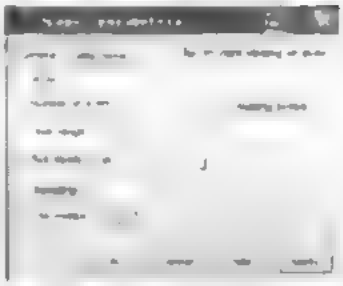


图9.25 设置"Scope"模块的属性

[illegible]

9.2.4 运行仿真系统

按照上面小节的步骤

Step 1 运行系统，[illegible]

图9.26 查看仿真结果

Step 2 [illegible]

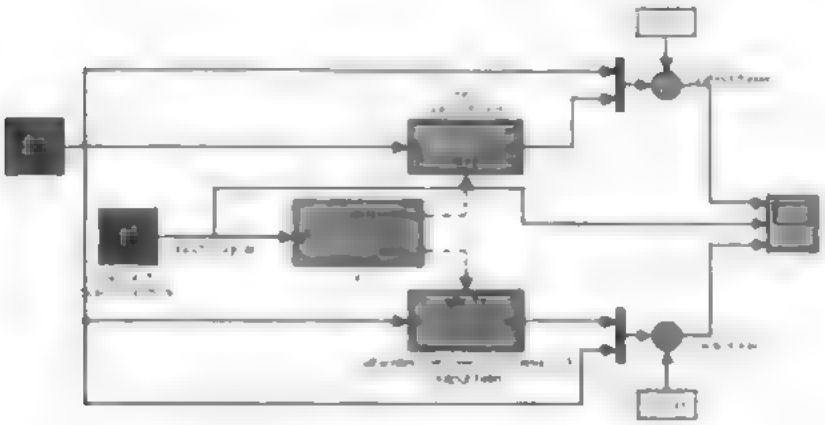


图 9.27 运行仿真后的系统模块

[illegible]成了点划线，表示封装的程序模块已经运行完成。

[illegible]

9.3 封装子系统

[illegible]独立于基础空间和其他模块库的模块空间。

[illegible]的事项。

9.3.1 封装子系统的创建方法

[illegible]

- [illegible]
- [illegible]
- 可以避免用户在描述操作中修改模块的参数数值。

在 Simulink 中，创建子系统的一般步骤如下

step 1 [illegible]

step 2 [illegible]，如图9.28所示。

step 3 [illegible]，关闭编辑器可以得到新建的封装子系统。

step 4 [illegible]上面的对话框，重新设置相应的属性。

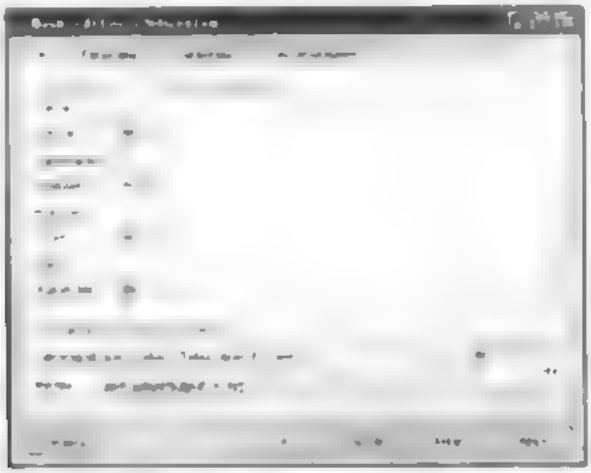


图9.28 封装编辑器

[illegible]

9.3.2 封装子系统的步骤

在本[illegible]，将系统中一个简单的实现[illegible]封装子系统，[illegible]，但是基本上涵盖了封装子系统的各个属性。下面分步骤详细介绍。

例9.4 [illegible]

step 1 添加子系统的基础模块。[illegible]的步骤中，设置封装子系统的属性，需要[illegible]子系统的模块，如图9.29所示。

图9.29 添加子系统的模块

step 2 设置[illegible]模块的属性，在[illegible]，变量m的数值是[illegible]封装子系统的参数属性，[illegible]m。[illegible]的步骤中将需要修改m的数值。[illegible]

step 3 封装子系统。[illegible]属性，[illegible]子系统，然后选择[illegible]，[illegible]，选择其中的[illegible]。[illegible]

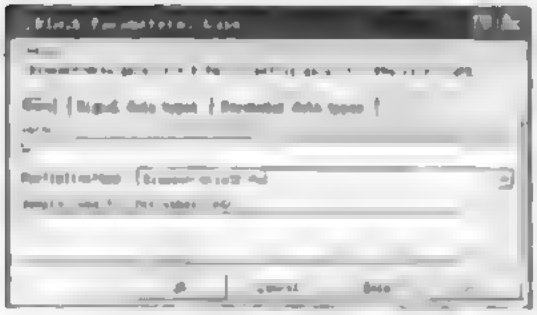


图 9.30 设置 Gain 模块的属性

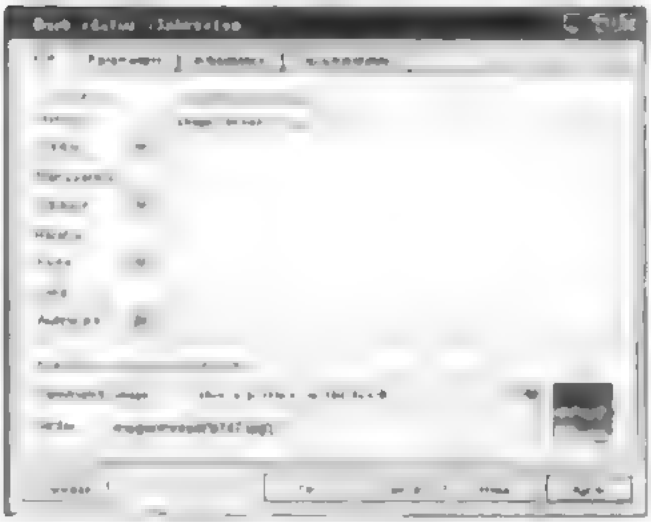


图 9.31 设置子系统的图标属性

- 在"Icon"选项卡的"Icon options"面板中，可以定义是否显示边框（Frame）[illegible]（Transparency）[illegible] Fixed [illegible] 等属性，[illegible]分析。
- 在"Drawing commands"选框中输入"image(imread('9.jpg'))"，表示将在 MATLAB 的当前路径中的"9.jpg"文件作为封装子系统的图标。
- 在"Examples of drawing commands"面板中，[illegible]模块[illegible]，[illegible]右侧出现的示例图标来编写图标绘制命令。

在 Simulink 中，[illegible] disp、fprintf、image、patch、[illegible]_label [illegible] 等[illegible]，参考"Simulink reference"中的"Mask Icon Drawing Commands"部分内容

step 6 设置封装子系统的"Parameters"参数。选择封装编辑器中的"Parameters"选项卡，在其中设置封装子系统的参数数值，如图 9.32 所示。

在上面的对话框中，单击"Add"按钮，开始添加子系统的参数，在"Prompt"选框中输入"Gain"，[illegible]提示文字。在"Variable"选框中输入"G"，表示[illegible]输入的参数值将传给变量 G。在"Type"选框中选择"edit"类型；最后，在"Dialog callback"选框中输入下面的程序代码

图9.32 设置封装子系统的参数

```
error('Gain is negative.')
```

error 中，

STEP 5 在其中设置封装子系统的参数初始数值，如图9.33所示。

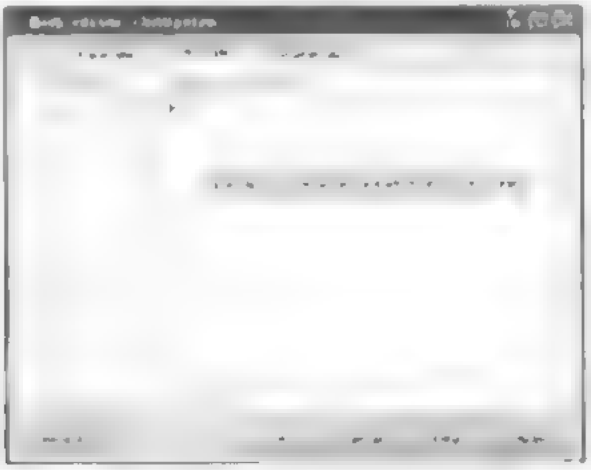


图9.33 设置子系统的参数初始数值

是初始化命令不能访问MATLAB工作空间中的变量。

STEP 6 封装子系统的说明文字，如图9.34所示。

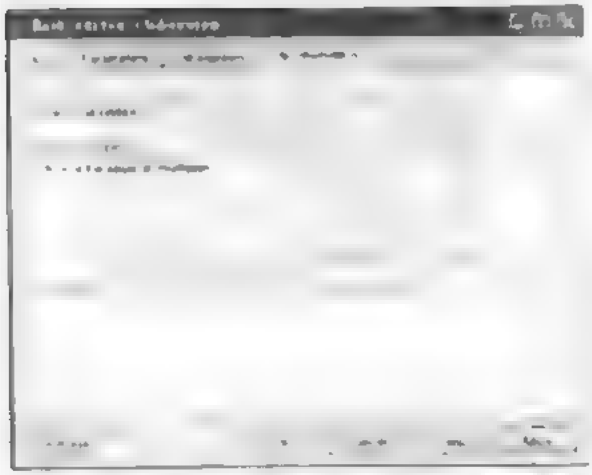


图 9.34 定义封装子系统的说明文字

[illegible]

[illegible]

step 7 查看封装结果。[illegible]"封装子系统子属性"[illegible]。单击"Apply"按钮或者"OK"按钮，就可以保存封装结果，如图 9.35 所示。

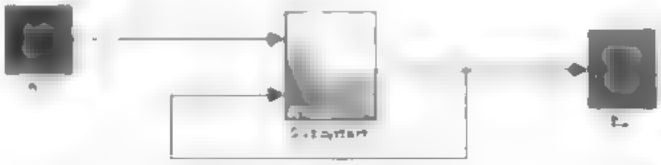


图 9.35 查看系统封装结果

step 8 设置封装子系统的参数数值。[illegible]，打开参数对话框，在其中输入变量 m 的数值，如图 9.36 所示。

图 9.36 设置封装子系统的参数数值

[illegible]"Parameters"[illegible]，[illegible]请框中填入变量 m 的数值。

step 9 [illegible]

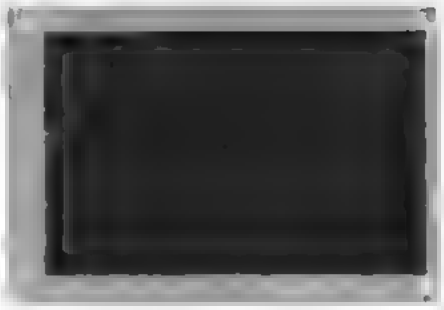


图9.37 查看仿真结果

step 10 [illegible]

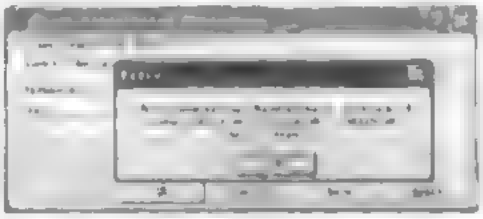


图9.38 系统提示错误信息

[illegible]

9.4 封装子系统实例

[illegible]

9.4.1 添加"Bang-Bang Controller"子系统

例9.5 [illegible] Anti-lock brake System，[illegible] when [illegible] 和车轮速度 wheel [illegible] angular speed [illegible] Sp1，控制是否运行汽车的刹车系统。下面详细介绍创建步骤。

step 1 [illegible] 等于0，则返回数值0。系统模块如9.39所示。

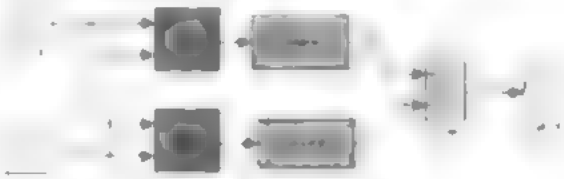


图9.39 添加"Bang-Bang Controller"模块

Step 2 [illegible]命令，如图 9.40 所示。

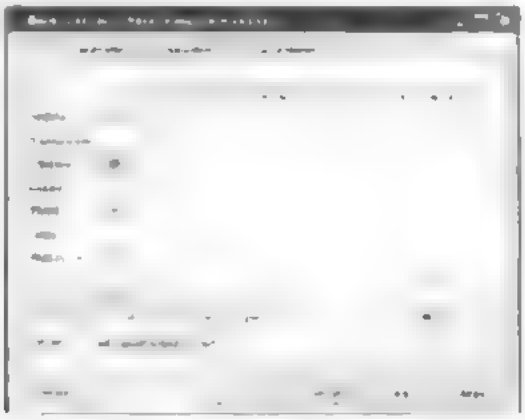


图 9.40 设置封装子系统的图标

在上面对话框中输入的绘图命令为

```
plot([-50,-50,50,50,[-50,50],[0,0],[0,0],[-50,50],[-40,0],[-30,-30],[0,
[illegible]
```

Step 3 [illegible]命令，得到的结果如图 9.41 所示。

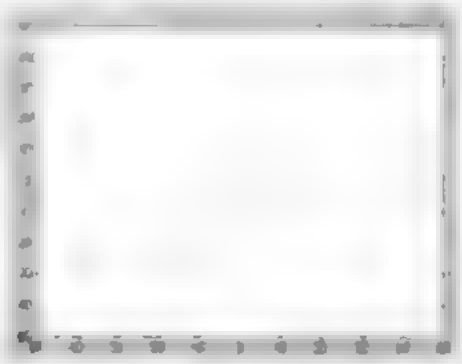


图 9.41 演示子系统的图标

Step 4 [illegible]子系统的说明文字，如图 9.42 所示。

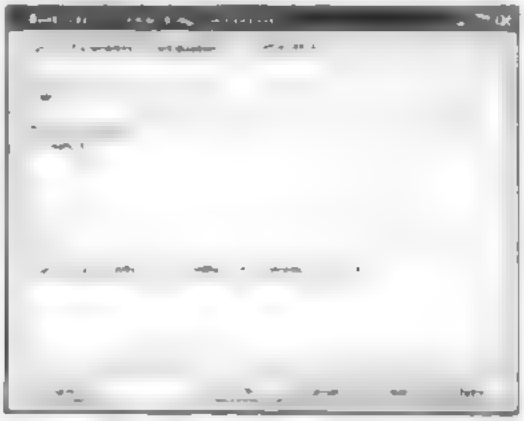


图 9.42 设置子系统的说明文字

9.4.2 添加“brake torque”子系统

继续上面小节的步骤

Step 1 [illegible]，如图9.43所示。

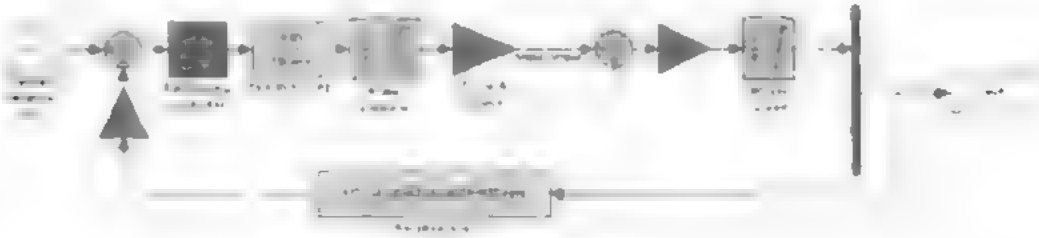


图9.43 添加系统模块

Step 2 [illegible]

- “Desired relative slip”模块：[illegible]的相对位移数值。
- 圆形求和（Sum）模块：[illegible]的相对位移模块的数值差值。
- “Ctrl”模块：[illegible]中Ctrl是系统的初始参数数值。
- “Relative Slip”模块：[illegible]角速度。该模块将需要添加新的系统模块来进行计算。
- “Bang-Bang controller”模块：[illegible]
- “Hydraulic Lag”模块：[illegible]初始参数的数值。
- “Brake pressure”模块：[illegible]分，得出系统的“刹车压力”数值。
- “Force &torque”模块：[illegible]

[illegible]

9.4.3 添加“tire torque”子系统

继续上面小节的步骤

Step 1 [illegible]

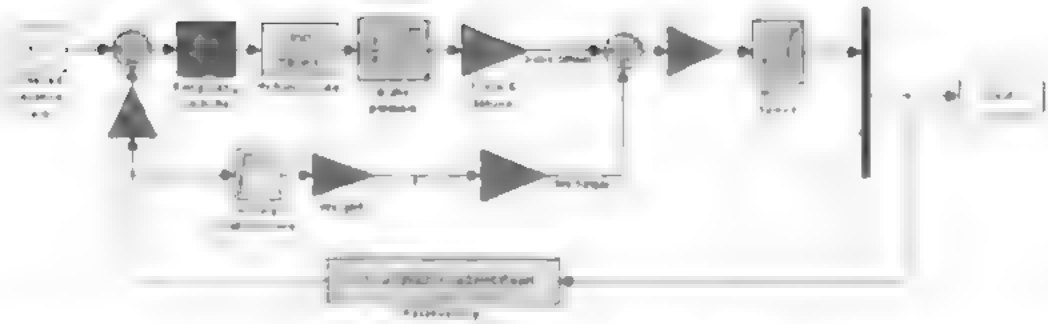


图 9.44 添加轮胎转矩的系统模块

step 2 双击 [illegible] 模块的属性，在弹出模块参数对话框中，[illegible] 数据模块，该模块的数据来源于 [illegible] 数据，输入数据为参数 [illegible]，[illegible] 系数 μ。该模块的属性对话框如图 9.45 所示。

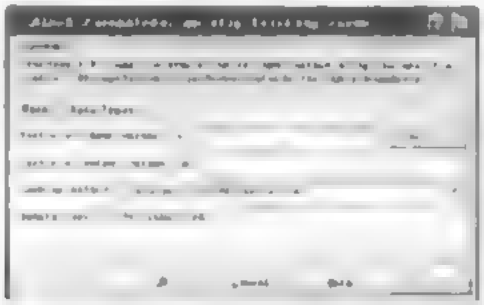


图 9.45 Lookup Table 模块的属性

step 3 在 [illegible] 模块的数据 [illegible] 选项卡中，[illegible] 数据，如图 9.46 所示。

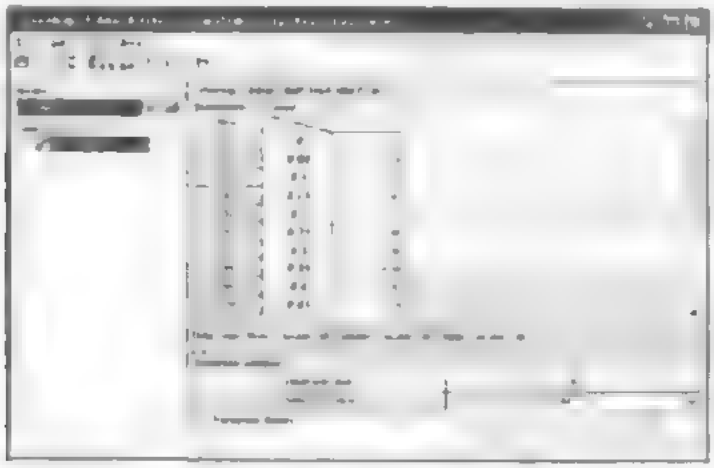


图 9.46 编辑模块的数据

"Lookup Table" 是 [illegible] 中的一个非线性模块，[illegible] "Lookup Table" 模块[illegible]，Simulink 模块库中的数学函数模块，[illegible] 方法。

step 4 [illegible]，其表达式为“m*g/4”，对应的参数对话框如图 9.47 所示。

图 9.47 “重量”模块数值

step 5 [illegible]

[illegible]

step 6 [illegible]

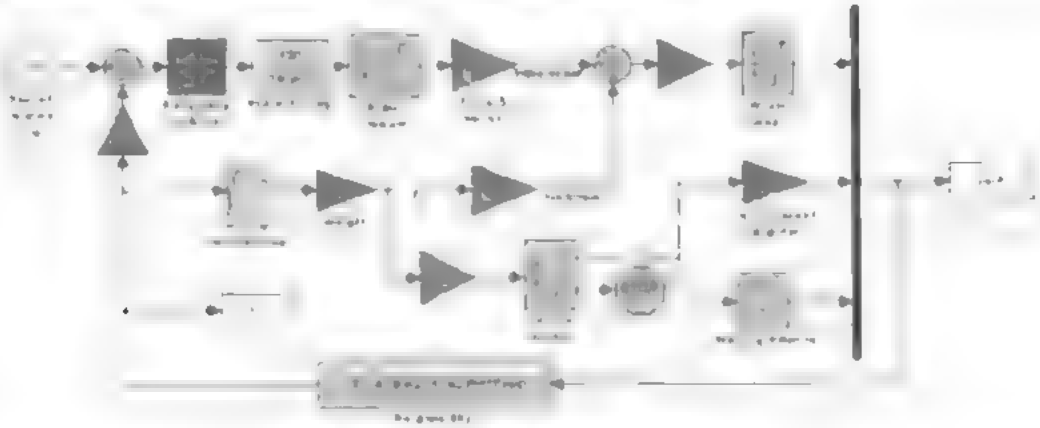


图 9.48 计算汽车运动的角速度

[illegible]

[illegible]

9.4.4 添加子系统的程序代码

[illegible]

step 1 [illegible]

入下面的代码

```
%显示加载数据的信息
fprintf('Loading data for ABS braking model...')
%定义常数信息
g = 32.18;
v0 = 88;
Rr = 15/12;  % Wheel radius
Kf = 1;
m = 50;
PBmax = 1500;
TB = 0.01;
I = 5;

%
% Mu slip 曲线
%
slip = (0:.05:1.0);
mu = [ 0 .4 .8 .97 1.0 .98 .96 .94 .92 .9 .88 .855 .83 .81 .79 .77 .75 ,
73 .72 .71 .7];
ctrl = 1;
%显示加载结束的信息
disp('done.');
```

将上面的所有数据保存为“Simdata.m”文件，在运行该仿真系统时，将需要首先加载上面文件中的数据。

step 2 编写运行仿真系统的代码。单击命令窗口工具栏中的D按钮，或者选择编辑栏中的“File”⇨“New”⇨“M-file”命令，打开M文件编辑器，在其中输入下面的代码：

```
%调用仿真文件
sim32
%设定仿真运行的时间
try
  time = sim('sim32',25);
catch
  simdata
  time = sim('sim32',25);
end
%创建图形对象
h = findobj(0, 'Name', 'ABS Speeds');
if isempty(h),
  h=figure('Position',[300   387   452   257],...
           'Name','ABS Speeds',...
           'NumberTitle','off');
end
figure(h)
set(h,'DefaultAxesFontSize',8)
%根据输出变量绘制图形
plot(time,yout(:,1:2))
%添加图形标题和坐标轴名称
title('Vehicle speed and wheel speed')
ylabel('Speed(rad/sec)')
xlabel('Time(secs)')
%设置图形的位置属性
set(gca,'Position',[0.1300    0.1500    0.7750    0.750])
```

```
%设置标题和坐标轴名称的字体大小
set(get(gca,'xlabel'),'FontSize',10)
set(get(gca,'ylabel'),'FontSize',10)
set(get(gca,'title'),'FontSize',10)

%添加带有箭头的注释文字
hold on
plot([5.958; 4.192],[36.92; 17.29],'r-',[5.758; 5.958; 6.029],[36.55;
36.92; 35.86],'r-' )
%设置文字内容
text(8.533,54.66,'Vehicle speed (\omega_v)','FontSize',10)
plot([7.14; 8.35],[43.1; 56.3],'r-',[7.34; 7.14; 7.07],[43.4; 43.1; 44.
1],'r-' )
text(4.342,15.69,'Wheel speed (\omega_w)','FontSize',10)
drawnow
hold off
%创建新的图形对象
h = findobj(0, 'Name', 'ABS Slip');
if isempty(h),
  h=figure('Position',[300    56   452   257],...
           'Name','ABS Slip',...
           'NumberTitle','off');
end
figure(h);
set(h,'DefaultAxesFontSize',8)
%根据变量绘制图形
plot(time,slp)
%添加图形标题和坐标轴名称
title('Slip')
xlabel('Time(secs)')
ylabel('Normalized Relative Slip')
set(gca,'Position',[0.1300    0.1500    0.7750    0.750])
set(get(gca,'xlabel'),'FontSize',10)
set(get(gca,'ylabel'),'FontSize',10)
set(get(gca,'title'),'FontSize',10)
```

将上面的代码保存为"runsim.m"文件，该文件将是运行该仿真的主要代码文件。关于该文件中的代码含义，请查看相应的关于图形的章节。

添加"Subsystem"子系统

延续上面小节的步骤。

step 1 添加新的"Subsystem"模块。选择"Simulink Library Browser"中"Commonly Used Blocks"模块库中的"Subsystem"模块，然后双击该模块，将子系统默认的数据输入和数据输出模块删除，得到的结果如图9.49所示。

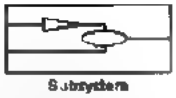


图9.49 添加新的Subsystem模块

step 2 封装上面的子系统。选中上面的子系统，单击鼠标右键，在弹出的快捷菜单中选择"Mask

[illegible]

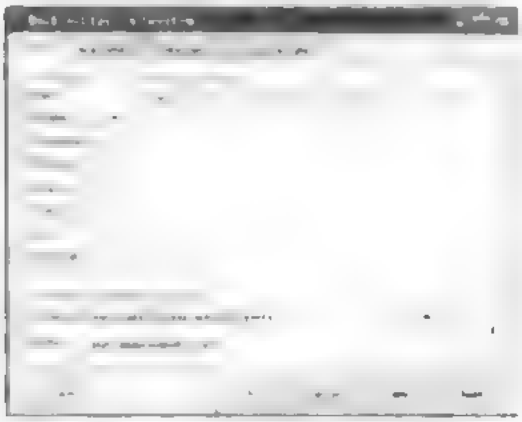


图 9.50 封装子系统

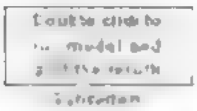
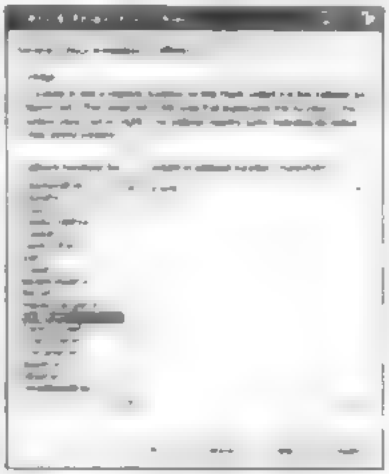


图 9.51 设置子系统的图标

step 3 [illegible]，得到的结果如图 9.52 所示。

[illegible]

step 4 [illegible]，如图 9.53 所示。



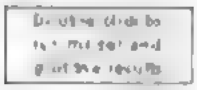


图 9.52 设置系统模块的外观属性

图 9.53 设置模块的属性

[illegible]，当双击该模块时，Simulink 将会调用 runsim 的 M 文件。

[illegible]

step 5 [illegible]

整系统模块，如图9.54所示。

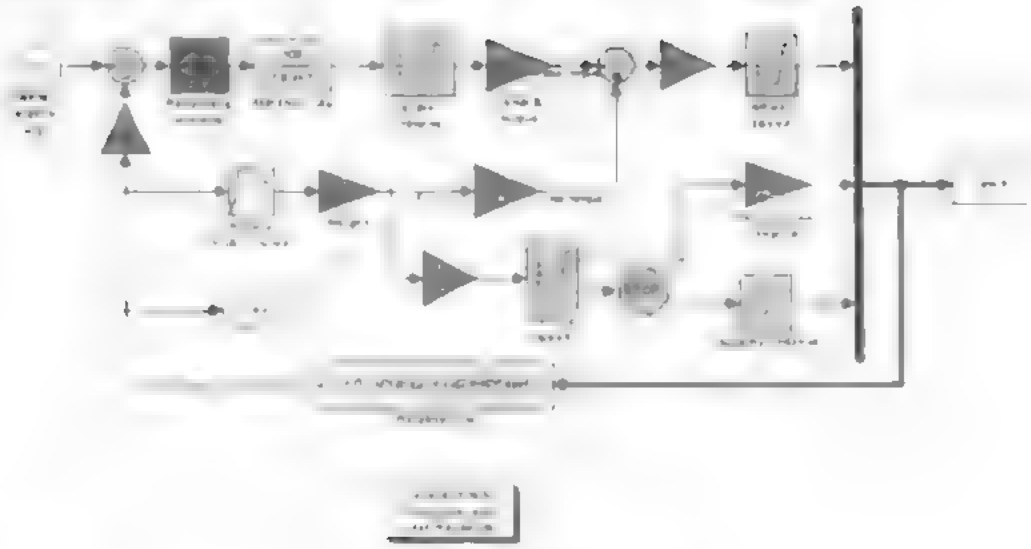


图9.54 完整的系统模块

9.4.6 运行仿真系统

继续上面小节的步骤。

Step 1 [illegible]

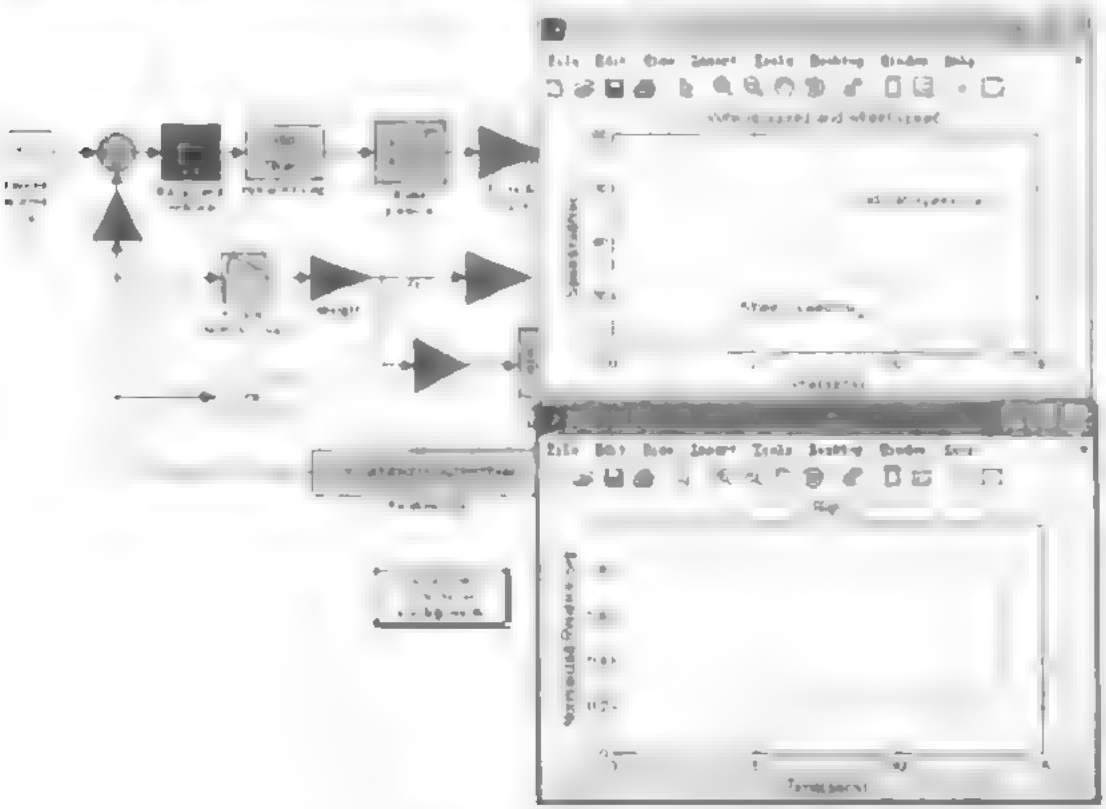


图9.55 查看仿真结果

Step 7 查看 MATLAB 命令窗口中数据[illegible]，[illegible]命令窗口，如图 9.56 所示。

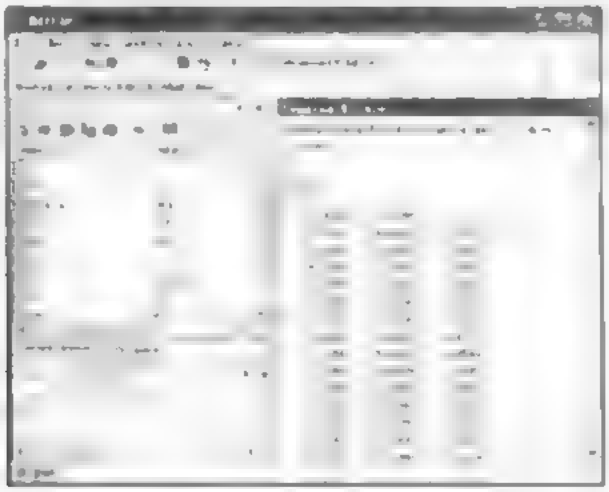


图 9.56 返回 MATLAB 命令窗口

在 MATLAB 命令窗口中，[illegible]。[illegible]

```
Loading data for ABS braking model...done
```

[illegible]

```
>> yout
yout =
   70.4000   70.4000         0
   70.4000   70.4000    0.0000
   70.4000   70.4000    0.0001
   70.4000   70.4000    0.0012
   70.4000   70.4000    0.0062
          ..........................................//限于篇幅，省略了部分数据
            0.6787  720.6747
            [illegible]  720.6829
            0.5886  720.6906
            0.5436  720.6977
            [illegible]  720.7042
            0.4535  720.7101
            0.4084  720.7155
            0.3634  720.7204
            0.3183  720.7246
            0.2733  720.7283
            [illegible]  720.7314
            [illegible]  720.7341
            0.1381  720.7362
            0.0931  720.7375
            0.0480  [illegible]
            0.0030  720.7387
            0.0000  720.7387
                 0  720.7387
```

9.5 使能（Enabled）子系统

9.5.1 创建使能子系统

例9.6

step 1

- **模块A：** Unit Delay模块。样本时间为0.25秒。
- **模块B：** Unit Delay模块。样本时间为0.5秒。
- **模块C：**
- **模块D：**

step 2

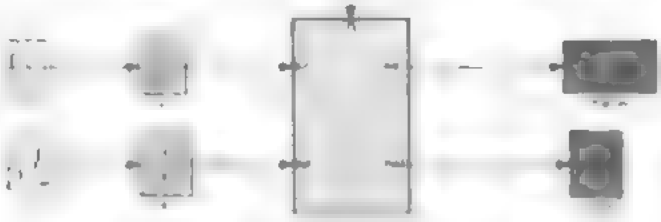


图9.57 添加系统模块

step 3 [illegible]

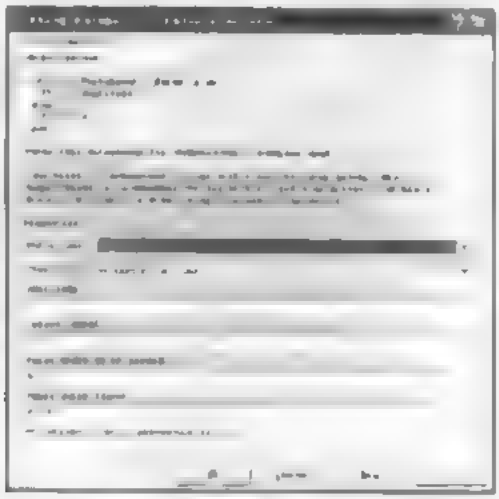


图 9.58 方形波的属性

step 4 [illegible] 9.59 所示。

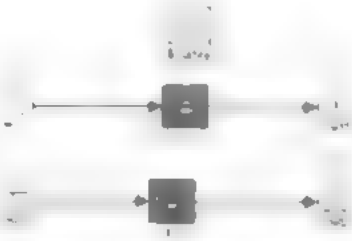


图 9.59 添加子系统模块

step 5 [illegible]

图 9.60 设置"Enable"模块的属性

step 6 [illegible] 钮，运行系统仿真，得到仿真结果，如图 9.61 所示。

[illegible] 运行情况，图 9.62 显示了使能子系统的启动时间。

9.5.2 使能子系统实例

[illegible]

子系统的属性，产生不合法的信号

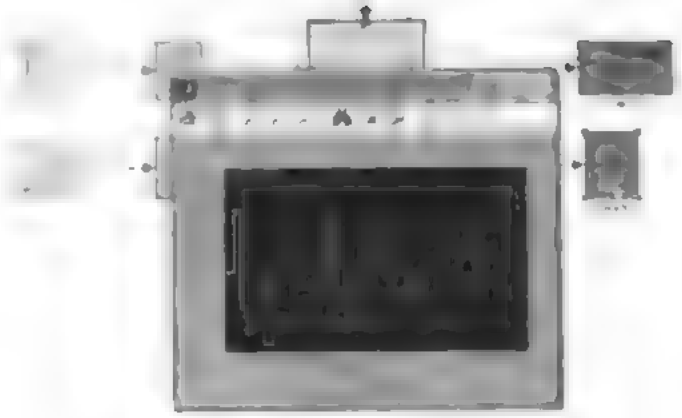


图 9.61　得到仿真结果

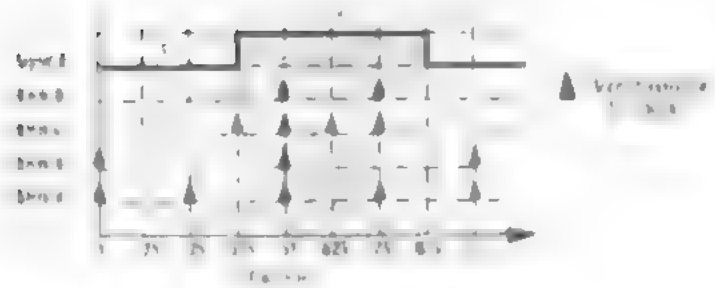


图 9.62　使能子系统的运行情况

例 9.7　创建一个使能子系统模块，该子系统根据[illegible]属性对同一个输入信号和控制信号产生不同的输出信号结果。

Step 1　[illegible]

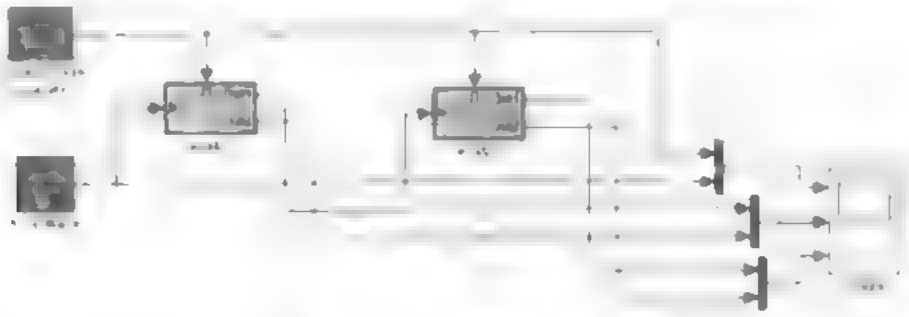


图 9.63　添加系统模块

Step 2　[illegible]属性，如图 9.64 所示。

Step 3　[illegible]，如图 9.65 所示。

图 9.64 设置输入信号的属性

图 9.65 查看变换函数的图形

step 5 [illegible]

图 9.66 设置控制信号的属性

[illegible]步骤中比较不同的信号特征。

Step 5 [illegible]子系统子模块，单击一面系统中的“[illegible]”模块，[illegible]子系统模块，如图 9.67 所示。

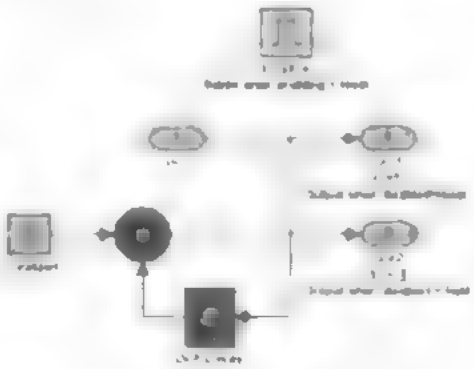


图 9.67 添加子系统模块

Step 6 [illegible]子系统的“Enable”模块属性，双击上面的“Enable”模块，[illegible]设置该模块的属性，如图 9.68 所示。

图 9.68 设置“Enable”模块的属性

[illegible]

Step 7 [illegible]子系统的“Out1”模块属性，单击[illegible]子系统中的“Out1”模块，单击[illegible]选择“[illegible]”选项，[illegible]设置该模块的属性，如图 9.69 所示。

图 9.69 设置输出模块的属性

在上面的对话框中的“Output when disabled”选项中选择“Reset”，这是该属性表明子系统中的“使能”子系统处在非工作状态时，输出信号使用的是“Reset”状态。同时，设置该输出模块的初始数值为 0，当系统恢复初始状态时，将保持数值为 0。

[illegible]

Step 8 点击"系统"[illegible]"模块的属性，选择上层子系统中的"Out1"模块，[illegible]，在弹出的菜单中选择"Outport Parameters"选项，打开对话[illegible]，在其[illegible]输出模块的属性，如图9.70所示。

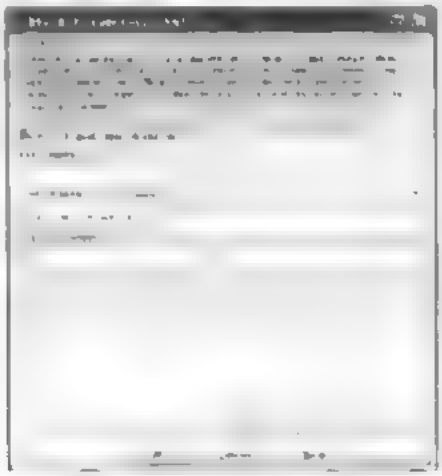


图9.70 设置输出模块2的属性

[illegible]

Step 9 [illegible]中添加该系统的模块，如图9.71所示。

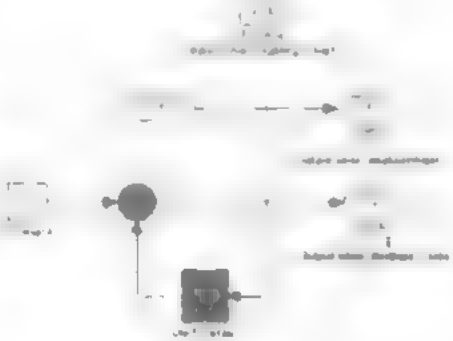


图9.71 添加第二子系统的模块

[illegible]第二个输出模块的状态属性是"held"，初始数值为0。

[illegible]

Step 10 至此，系统仿真。单击模块编辑器上的“开始仿真”按钮，运行整个系统，得到仿真结果如图 9.72 所示。

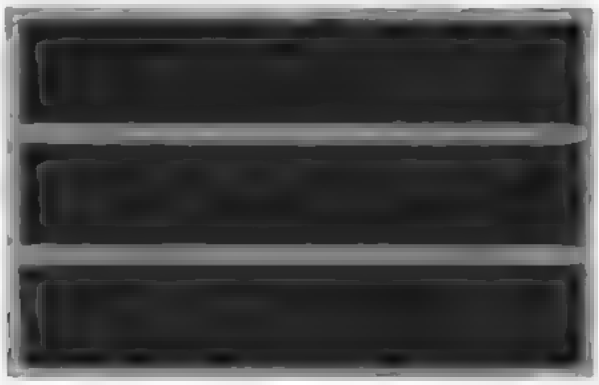


图 9.72 音乐系统仿真的结果

[illegible]

9.6 触发（Triggered）子系统

[illegible]

9.6.1 触发子系统简介

可以设置三种类型的触发事件来触发不同的子系统，各种触发事件的情况如下：

- rising：[illegible]
- falling：[illegible]
- either：当系统发生上述两种情况中任何一种情况，都会触发子系统。

[illegible]

[illegible]

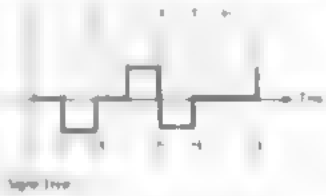


图 9.73 高敏系统的时序图表

[illegible]升之前，仅保存一个时间段的零值，因此不能触发子系统。

9.6.2 触发子系统的属性

[illegible]属性内容，将分步骤详细介绍。

例 9.8 [illegible]

Step 1 [illegible]，其对应的属性如图 9.74 所示。

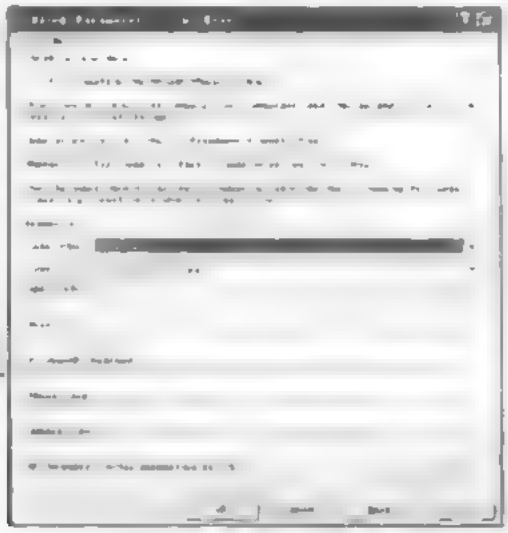


图 9.74 系统输入信号的属性

[illegible]

Step 2 [illegible]，其对应的属性如图 9.75 所示。

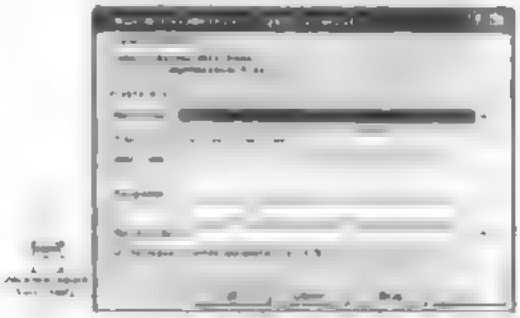


图 9.75 系统控制信号的属性

[illegible]

同时在"Fregency"选项中输入"1"。在"Units"选项中选择"Hertz"作为频率单位，因此也就设置了该方形波的频率为1Hz。

[illegible]

step 3 [illegible]信号，可以添加对应的系统模块，如图9.76所示。

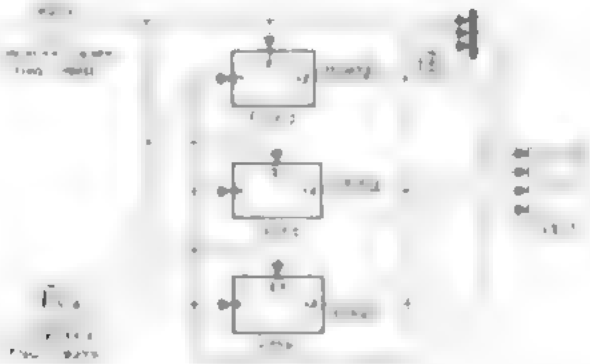


图9.76 添加系统模块

[illegible]

step 4 [illegible]

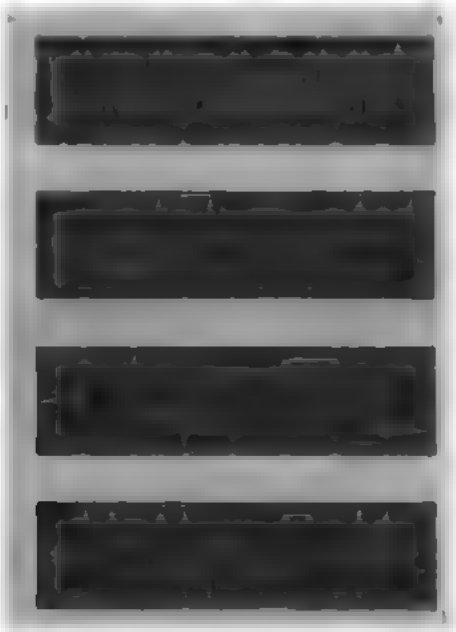


图9.77 查看不同的输出信号

[illegible]

[illegible]

[illegible]

9.7 触发子系统实例

[illegible]

9.7.1 添加系统模块

例 9.9 [illegible]

step 1 添加整个机械的发动机运行模块，模型的结构如图 9.78 所示。

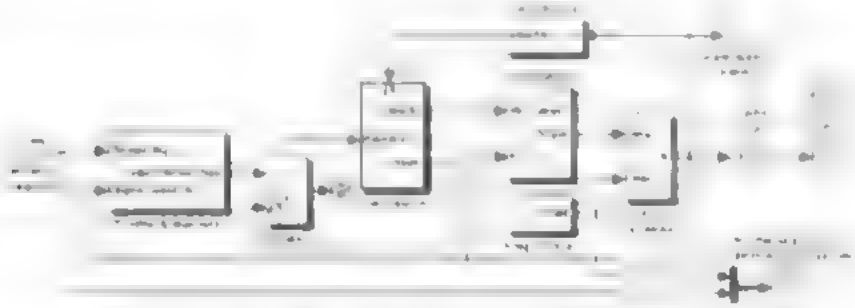


图 9.78 整个机械运行的系统模块

[illegible]

[illegible]

Step 2 设置"[illegible]"模块的属性。在建模过程中，"[illegible]"模块的输出信号[illegible]，以属性编辑器的"[illegible]"选项[illegible]，然后保持不变，其对应的属性对话框如图9.79所示。

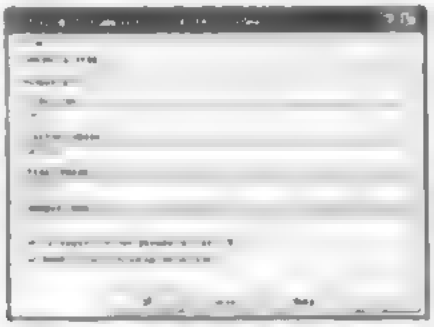


图9.79 设置输出信号的属性

提示 [illegible]

9.7.2 设置"Throttle & Manifold"子系统属性

继续上面小节的步骤。

Step 1 双击"[illegible]"子系统模块，[illegible]"Throttle & Manifold"模块，在[illegible]模块编辑器中，[illegible]子系统模块，如图9.80所示。

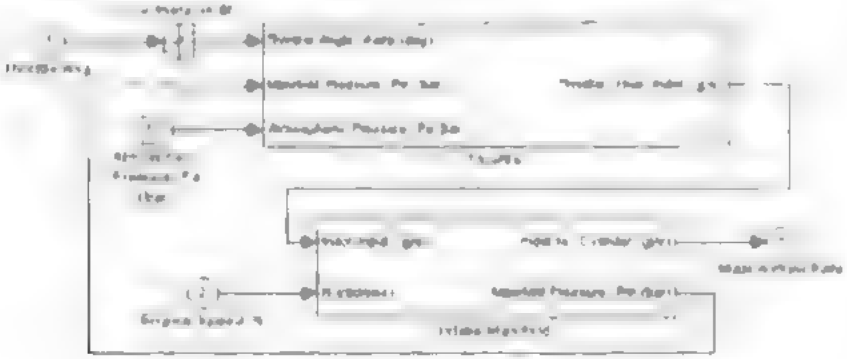


图9.80 设置"Throttle & Manifold"子系统模块

[illegible]子系统模块的[illegible]通过两个输入信号变量"Throttle Angle"和"Engine Speed"，[illegible]，输出变量"Mass Airflow rate"，其[illegible]。通过其中的子系统变量"Throttle Angle"，[illegible]"Manifold Pressure"[illegible]和"Atmospheric Pressure"[illegible]，通过计算得到变量"Throttle Flow"[illegible]。外[illegible]计算得到的变量和"Engine Speed"[illegible]，[illegible]子系统模块的输出变量"Mass Airflow rate"[illegible]"Manifold Pressure"[illegible]。

Step 3 设置"Saturation"模块的属性。"Saturation"模块的[illegible]控制输入信号的幅度范围，其属性如图9.81所示。

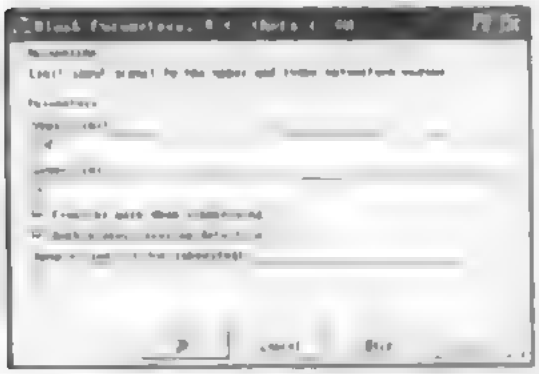


图 9.81 设置控制信号的属性

step 4 在[illegible]算变量"throttle-flow"[illegible]的子系统模块，在其[illegible]throttle-flow"下的清体 I 的模块，在其中添加子系统的模块如图 9.82 所示。

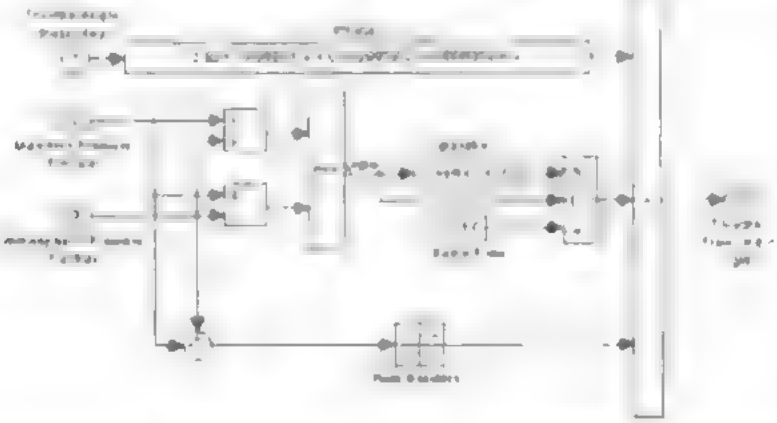


图 9.82 计算中间参量的子系统模块

step 5 [illegible]子系统模块的[illegible]，在子系统模块中，首先在两个端口输入 $f(\theta_i)$ 算出一个参数数值 δ_1，其中 $f(\theta_i)$ 的计算结果是 $f(\theta_i)=2.282\times10^{-4}(0.231\theta_i)^{3}-0.002990\theta_i^{2}+0.000630\theta_i$[illegible]，计算变量 Manifold Pressure（支管压力）与 Atmospheric Pressure（大气压力）的数值比例 λ，通过标准模块保证 $\lambda<1$，然后与 λ 比较，中间参数数值 δ 等于常数 Sonic Flow 与 λ 大小中[illegible]，将结果与 $2.\lambda-\lambda^{2}$ 的[illegible]中间参数数值 δ_2。最后，通过"signum"模块，计算[illegible]，当支管压力大于大气压力的时候，选择正号；当支管压力小于大气压力的时候，选择负号。

在子系统外模块，将两个中间参数相乘得到变量"throttle-flow"的数值，通过计算，产生一个[illegible]公式：Throttle-Flow=signum($Pm-Pa$)×δ_1×δ_2。

提示

在上面的分析过程中，各个变量模块的参数计算是非常重要的，关于这个系统公式的由来和具体的意义含义，由于篇幅有限，在这里不展开。读者可以查阅相关资料，了解具体的含义。本书以在计算过程中对中间参量的含义，是为了更清楚地理解模块的功能。

step 6 在[illegible]算输出变量"Air to Cylinder"[illegible]变量"Manifold Pressure"[illegible]子系统模块，添加的系统模块如图 9.83 所示。

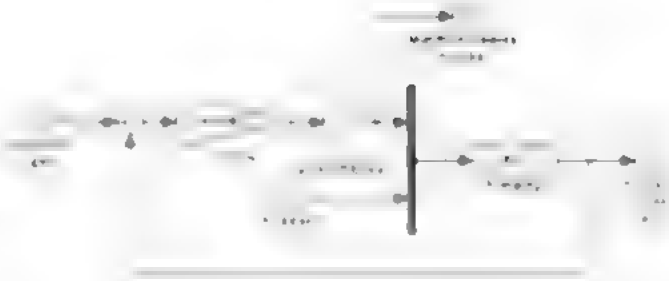


图 9.83　计算中间参数的子系统模块

step 1 [illegible] 方程组

$$\begin{cases} 0.08979x_1 - 0.01337x_1 x_2 + 0.00016x_1^2 + [illegible] \\ 0.41328(T - F - y) = x' \\ x(0) = 0.543 \end{cases}$$

[illegible] Throttle-Flow 数值，其中 0.41328 是普朗特转换系数。

[illegible]

9.7.3　设置“Intake”子系统属性

接续上面小节的步骤，

step 1 设置“Intake”模块的属性。在上一步骤中，“Intake”模块，设置完成了输入属性，如图 9.84 所示。

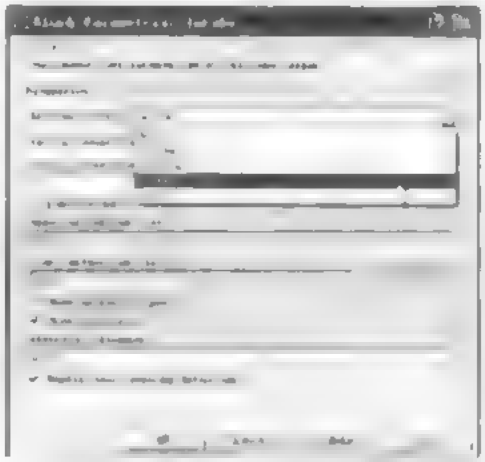


图 9.84　设置积分器模块的属性

Step 2 [illegible]

[illegible]

9.7.4 设置“Compression”子系统属性

接续上面小节的步骤。

Step 1 设置“Compression”子系统模块属性。[illegible]“Compression”模块，[illegible]后的系统模块。如图 9.85 所示。

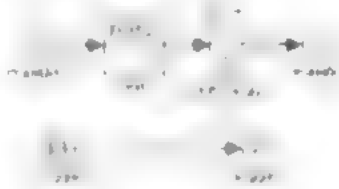


图 9.85 “Compression”子系统模块

[illegible]

Step 2 设置“Compression”模块属性。[illegible]“Compression”模块[illegible]属性对话框，设置整个系统的触发类型，如图 9.86 所示。

图 9.86 设置触发器类型

9.7.5 设置“Combustion”子系统属性

接续上面小节的步骤。

Step 1 设置“Combustion”子系统模块属性。[illegible]“Combustion”模块，[illegible]后的系统模块。如图 9.87 所示。

图 9.87 设置"Combustion"子系统模块

Step 2 [illegible]

扭矩的第一部分数据 δ_1

$$-181.3+370.36x_1+21.91x_2^2-0.85x_1^3-0.26x_1-0.0028x_1$$

[illegible]

动机相矩的第二部分的数据 δ

$$0.027x_3-0.000107x_3^2+0.00048x_1x_3+2.55x_1x_4-0.05x_1x_4^2$$

[illegible]

$$\delta=\delta_1+\delta$$

[illegible]

9.7.6 设置"Drag Torque"子系统属性

[illegible]

Step 1 [illegible]

[illegible]

图 9.88 设置"Drag Torque"子系统的模块

[illegible]

- 跳跃时间 25，初始数值 25，终止数值 20。
- 跳跃时间 85，初始数值 0，终止数值 5。

Step 2 [illegible]

信号叠加后的结果，如图 9.89 所示。

图 9.89 阻力矩的信号波形

9.7.7 设置"Vehicle Dynamics"子系统属性

[illegible]

Step 1 设置"Vehicle Dynamics"子系统的模块。双击上面系统模块"Vehicle Dynamics"模块，添[illegible]系统模块，[illegible]

图 9.90 设置"Vehicle Dynamics"子系统的模块

Step 2 设置"Vehicle Dynamics"子系统的功能。在上面的系统模块中，将[illegible]计算得到的[illegible]重要的动力矩，然后将其和[illegible]相乘再经过积分，得到发动机的速度 N_e。

[illegible]

9.7.8 设置"valve timing"子系统属性

[illegible]

Step 1 设置"valve timing"子系统模块。[illegible]"valve timing"模块，[illegible]的系统模块。如图 9.91 [illegible]

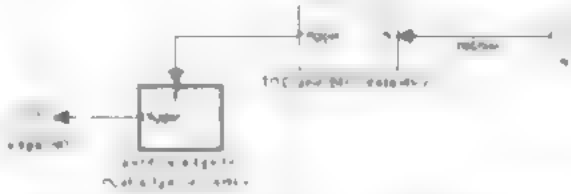


图 9.91 设置"valve timing"子系统的模块

在上面的程序模块中，首先通过输入端口输入发动机速度进行 TDC（上死点）和 BDC（下死点）的检测，输出触发信号[illegible]，然后设置触发器模块的属性，得出输出信号。

Step 2 [illegible]

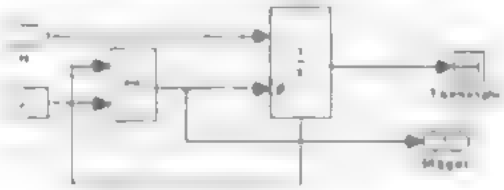


图 9.92 “TDC and BDC detection”模块

[illegible]

Step 3 [illegible]积分器属性如图 9.93 所示。

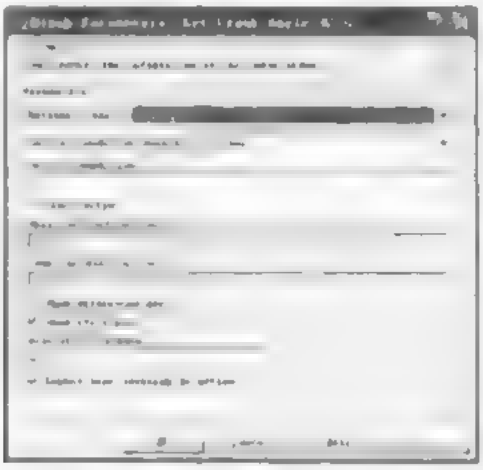


图 9.93 设置积分器的属性

Step 4 [illegible]

图 9.94 设置子系统模块

9.7.8 运行仿真系统

继续上面小节的步骤。

Step 1 [illegible]真结果，如图 9.95 所示。

图 9.95 发动机速度

step 2 [illegible]

9.8 S 函数（S-Function）

[illegible]

9.8.1 S 函数概述

在 Simulink 中，S 函数采用一种特殊的调用语法使得函数可以和 Simulink 方程解法器进行交互，这种形式的交互和解法器和 Simulink 系统自身提供的模块之间的交互十分相似。S 函数的形式比较通用，可以使用 S 函数来描述连续、离散和混合系统。

一般而言，S 函数可以使用在下面的场合：

- 生成用户自行研究的算法或算法调用的 S 函数模块
- 可以用来代表硬件驱动的模块
- 可以通过 S 函数将某个系统描述成一组数学方程组
- 编辑用于图形动画表现的 S 函数模块

[illegible]

[illegible]

9.8.2 S 函数的运行机理

[illegible]

[illegible]

Simulink 的仿真流程如图 9.96 所示。

图 9.96　Simulink 的仿真流程

[illegible]

[illegible]

[illegible]

9.8.3 S 函数模板

[illegible]

```
function [sys,x0,str,ts] = sfuntmpl(t,x,u,flag)
%[illegible]
%关于该函数的输入变量名称、数目和次序，一般情况下不要改变
[illegible]
```

```
%在上面的参数中，flag是标记变量，共有6个不同的取值，分别代表6个不同的子函数

switch flag,
  case 0,
    [sys,x0,str,ts]=mdlInitializeSizes;              %调用初始化的子函数
  case 1,
    sys=mdlDerivatives(t,x,u);                       %调用计算模块导数的子函数
  case 2,
    sys=mdlUpdate(t,x,u);                            %调用更新模块离散状态的子函数
  case 3,
    sys=mdlOutputs(t,x,u);                           %调用计算模块输出的子函数
  case 4,
    sys=mdlGetTimeOfNextVarHit(t,x,u);               %调用计算下一个采样时点的子函数
  case 9,
    sys=mdlTerminate(t,x,u);                         %调用结束仿真的子函数
  otherwise
    error([ 'Unhandled flag = ',num2str(flag)]);
end
%=============================================================================%
function [sys,x0,str,ts]=mdlInitializeSizes
sizes = simsizes;              %调用simsizes函数，返回规范格式的sizes构架
                               %这是一个通用的函数语句，用户不要轻易修改
sizes.NumContStates  = 0;      %计算系统模块的连续状态数目，0是默认数值
                               %用户应该对自己创建的系统进行修改
sizes.NumDiscStates  = 0;      %计算系统模块的离散状态数目，0是默认数值
                               %用户应该对自己创建的系统进行修改
sizes.NumOutputs     = 0;      %计算系统模块的输出数目，0是默认数值
                               %用户应该对自己创建的系统进行修改
sizes.NumInputs      = 0;      %计算系统模块的输入数目，0是默认数值
                               %用户应该对自己创建的系统进行修改
sizes.DirFeedthrough = 1;      %计算系统模块中直接通向返回路线的数目，1是默认数值
                               %用户应该对自己创建的系统进行修改
sizes.NumSampleTimes = 1;      %计算系统模块中采样时间的数目，1是默认数值
                               %用户应该对自己创建的系统进行修改
sys = simsizes(sizes);         %初始化后的构架sizes经过simsizes函数运算后向sys赋值
                               %这是系统默认的命令，用户不要轻易修改
x0  = [];                      %向模块的初始值赋值，其中[]是默认数值
                               %用户应该对自己创建的系统进行修改
str = [];
ts  = [0 0];
%=============================================================================%
function sys=mdlDerivatives(t,x,u)
%编写计算导数向量的命令
sys = [];
%=============================================================================%
function sys=mdlUpdate(t,x,u)
%编写计算更新模块离散状态的命令
sys = [];
%=============================================================================%
function sys=mdlOutputs(t,x,u)
%编写计算模块输出向量的命令
sys = [];
%=============================================================================%
function sys=mdlGetTimeOfNextVarHit(t,x,u)
%该函数只有在"变采样时间"条件下使用
sampleTime = 1;
```

```
sys = t + sampleTime;
%=============================================================================%
function sys=mdlTerminate(t,x,u)
sys = [];
```

在上面的程序代码模板中，多次引用系统函数simsizes，默认情况下，其保存路径的目录为MATLAB7.0\toolbox\simulink\simulink。该函数的主要目的在于设置某个S函数的大小，具体的函数程序代码如下：

```
function sys=simsizes(sizesStruct)
switch nargin,
  case 0,  % return a sizes structure
    sys.NumContStates  = 0;
    sys.NumDiscStates  = 0;
    sys.NumOutputs     = 0;
    sys.NumInputs      = 0;
    sys.DirFeedthrough = 0;
    sys.NumSampleTimes = 0;
  case 1,  % convert a sizes structure into an array, or the other way around
      if ~isstruct(sizesStruct),
      sys = sizesStruct;
      if length(sys) < 6,
        error('Length of sizes array must be at least 6');
      end
      clear sizesStruct;
      sizesStruct.NumContStates  = sys(1);
      sizesStruct.NumDiscStates  = sys(2);
      sizesStruct.NumOutputs     = sys(3);
      sizesStruct.NumInputs      = sys(4);
      sizesStruct.DirFeedthrough = sys(6);
      if length(sys) > 6,
        sizesStruct.NumSampleTimes = sys(7);
      else
        sizesStruct.NumSampleTimes = 0;
      end
    else,
      % validate the sizes structure
      sizesFields=fieldnames(sizesStruct);
      for i=1:length(sizesFields),
        switch (sizesFields{i})
          case { 'NumContStates', 'NumDiscStates', 'NumOutputs',...
                 'NumInputs', 'DirFeedthrough', 'NumSampleTimes' },
          otherwise,
            error([ 'Invalid field name ''', sizesFields{i}, '''']);
        end
      end
      sys = [ ...
        sizesStruct.NumContStates,...
        sizesStruct.NumDiscStates,...
        sizesStruct.NumOutputs,...
        sizesStruct.NumInputs,...
        0,...
        sizesStruct.DirFeedthrough,...
        sizesStruct.NumSampleTimes ...
      ];
```

[illegible]

下面对模板函数代码进行简单的说明。

[illegible]

- t 表示当前时刻，采用单精度的时间数值
- x 表示模块的状态向量
- u 表示模块的输入向量
- flag 程序的标记变量，对应不同的操作类型。

其对应的状态过程模块如图 9.97 所示。

图 9.97 系统的状态过程模块

9.8.4 添加 S 函数模块

[illegible]

例 9.10 创建一个简单的实例，说明如何创建 S 函数。

Step 1 [illegible] 块将作为输入信号，如图 9.98 所示。

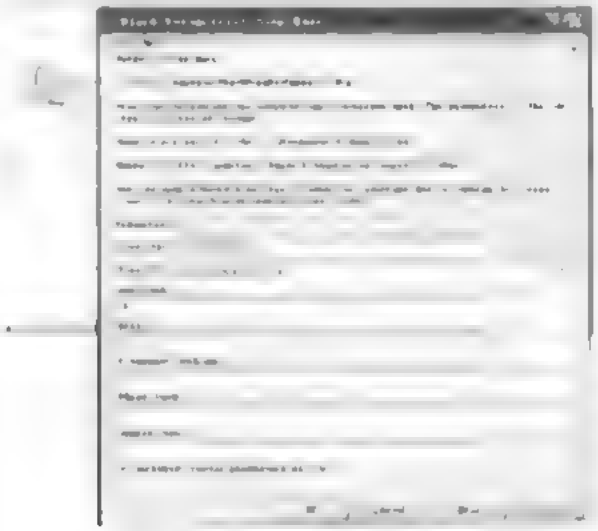


图 9.98 添加输入信号模块

step 2 [illegible] file S-Function"，如图 9.99 所示。

图 9.99　添加 S 函数模块

step 3 [illegible] "Mask S-function" 选项，如图 9.100 所示。

图 9.100　封装 S 函数模块

[illegible] 封装子系统的图标，如图 9.101 所示。

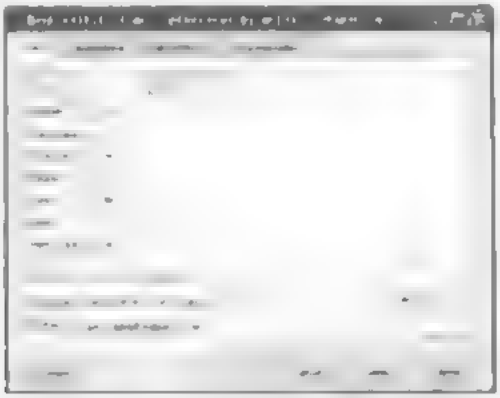


图 9.101　设置封装子系统的图标

step 4 [illegible] 参数值，如图 9.102 所示。

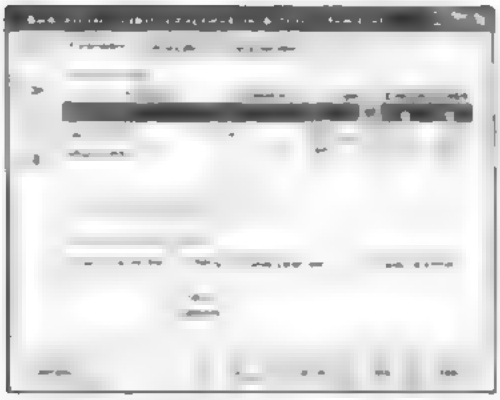


图 9.102 设置仿真模块的参数

[illegible]

step 5 设置仿真系统的说明文字。选择上面对话框中的"Documentation"选项卡，在其中设置封装子系统的说明文字，如图 9.103 所示。

图 9.103 设置仿真系统的说明文字

step 6 查看封装子系统模块[illegible]，系统模块[illegible]。

图 9.104 完成后的系统模块

9.8.5 添加 S 函数程序代码

接续上面小节的步骤。

step 1 设置仿真系统的子函数参数。选择上面模块中的子函数模块，单击鼠标右键，在弹出的快捷菜单中选择"S-function Parameters"选项，如图 9.105 所示。

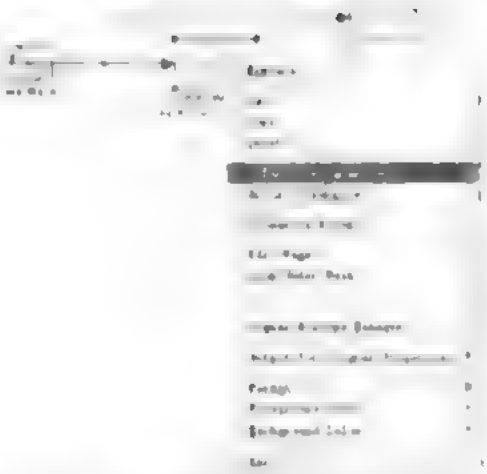


图 9.105　设置仿真系统的函数参数

[illegible]

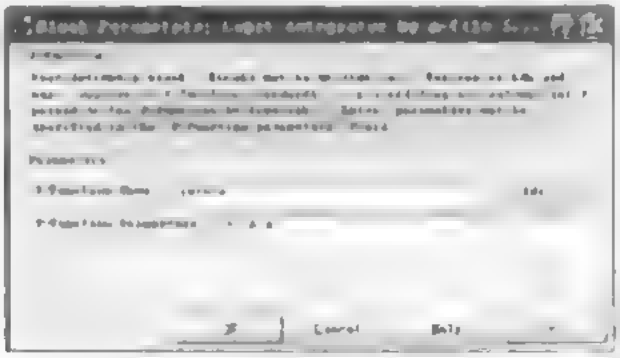


图 9.106　S 函数的参数设置

Step 2 [illegible]

按钮，就可以查看对应的 S 函数，其代码如下

```
function [sys,x0,str,ts] = [illegible](t,x,u,flag,lb,ub,xi)
%计算表达式的布斯积分数值
%判断参数 flag 的数值
switch flag
  case 0
    [sys,x0,str,ts] = mdlInitializeSizes(lb,ub,xi);
                          %调用mdlInitializeSizes函数
  case 1
    sys = mdlDerivatives(t,x,u,lb,ub);    %调用mdlDerivatives函数
  case {2,9}
    sys = [];       %不进行任何的操作
  [illegible]
    sys = mdlOutputs(t,x,u);    %调用mdlOutputs函数
  otherwise
    error(['unhandled flag = ',num2str(flag)]);    %显示错误信息
end
%=============================================================
function [sys,x0,str,ts] = mdlInitializeSizes(lb,ub,xi)
sizes = simsizes;
```

```
sizes.NumContStates  = 1;
sizes.NumDiscStates  = 0;
[illegible]          = 1;
sizes.NumInputs      = 1;
sizes.DirFeedthrough = 0;
sizes.NumSampleTimes = 1;
[illegible]
str = [];
x0  = x;
ts  = [0 0];   % 样本时间
%=============================================================================
function sys = mdlDerivatives(t,x,u,lb,ub)
if (x <= lb & u < 0)  | (x>= ub & u>0 )
  sys = 0;
[illegible]
  sys = u;
end
%=============================================================================
function sys = mdlOutputs(t,x,u)
sys = x;  % 返回输入参数x的数值
```

[illegible]

step 1 [illegible]分参数，如图9.107所示。

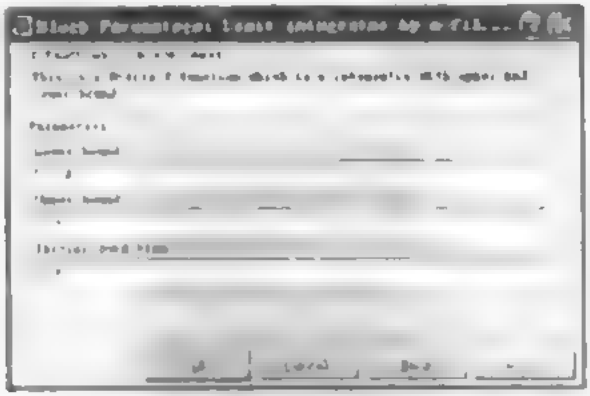


图9.107 设置积分的上下限

[illegible]

9.8.6 运行仿真

继续上面小节的步骤。

step 1 [illegible]果如图9.108所示。

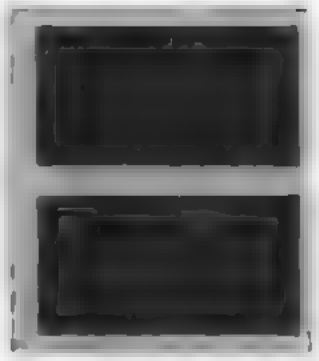


图9.108 得到仿真结果

step 2 修改系统仿真参数，重新[illegible]，[illegible]，在其中重新设置系统参数，[illegible]

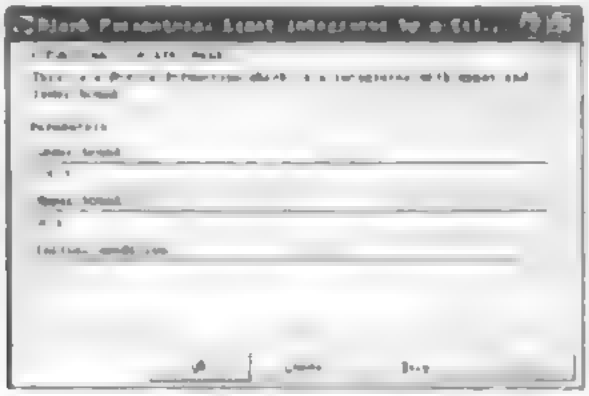


图9.109 重新设置仿真参数

step 3 重新[illegible]，单击[illegible]按钮，然后重新[illegible]仿真，结果如图9.110所示。

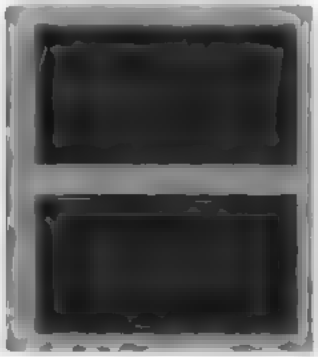


图9.110 新的仿真结果

9.9 S函数实例

本节将介绍如何在Simulink中使用S函数来模拟动态物体和弹簧系统[illegible]，首先根据物理知识列出物体的状态微分方程，然后根据微分结果设置参数编写[illegible]函数，[illegible]实现动画的演示[illegible]时。本实例还将涉及到读取数据、状态分析等关于Simulink的[illegible]

9.9.1 添加系统模块

例 9.11 创建两个物体和弹簧系统满足要求的仿真模型。

step 1 添加系统模块，[illegible]，然后分步骤详细介绍每个模块的属性，如图 9.111 所示。

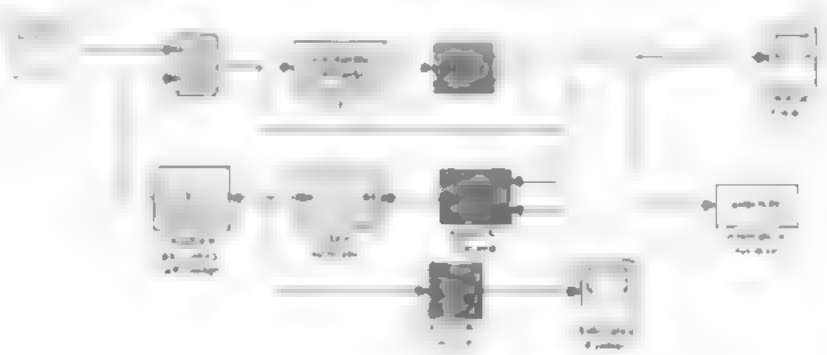


图 9.111 添加系统模块

step 2 设置输入信号的属性，在本文例中，输入信号为方波，代表三相交流电的一个周期，其具体的属性如图 9.112 所示。

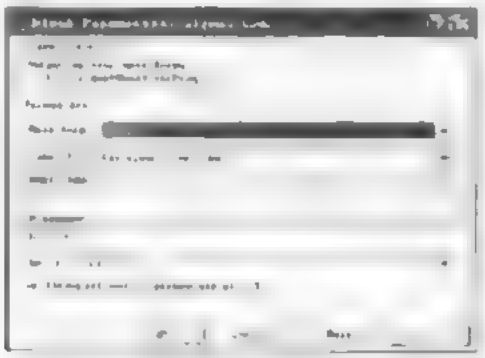


图 9.112 方波的属性

step 3 设置"Plant"模块的属性，在本文例中，"Plant"模块属于"[illegible]"模块库，代表该系统适应的微分方程组，其属性如图 9.113 所示。

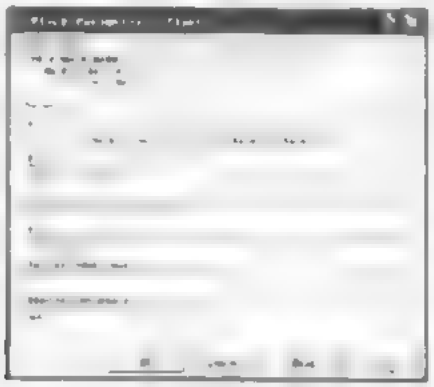


图 9.113 设置"Plant"模块的属性

step 4 分析"Plant"模块的原理，该模块[illegible]，[illegible]

[illegible]

$$
\begin{cases}
\dfrac{\mathrm{d}x_1(t)}{\mathrm{d}t} = -\dfrac{Ka}{m_1}x_1(t) + \dfrac{Ka}{m_1}x_3(t) + u \\
\dfrac{\mathrm{d}x_2(t)}{\mathrm{d}t} = x_1(t) \\
\dfrac{\mathrm{d}^2x_3(t)}{\mathrm{d}t^2} = \dfrac{Ka}{m_2}x_1(t) - \dfrac{Ka}{m_2}x_3(t) \\
\dfrac{\mathrm{d}x_3(t)}{\mathrm{d}t} = x_4(t)
\end{cases}
$$

在一[illegible]中，变量 $x_1(t)$、$x_2(t)$ [illegible] m_1、m_2 [illegible]，$x_1(t)$、$x_3(t)$ [illegible]，[illegible] u 表[illegible]形成输入信号，另外一个部分则是自反馈的信号。

然后根据下一个方程 $y=Cx+Du$，得到的关系等式如下

$$
y = \begin{pmatrix} x_2(t) \\ x_3(t) \\ x_4(t) \end{pmatrix}
$$

[illegible]

[illegible]

Step 5 设置“Dmux”模块的属性。[illegible]，“Dmux”模块[illegible]进行分解，其属性如图 9.114 所示。

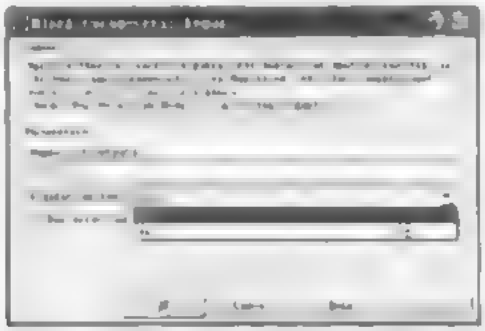


图 9.114 设置“Dmux”模块的属性

在一[illegible]“Display port sizes”[illegible]“Dmux”，[illegible]“Port sizes”[illegible]，[illegible]和一个一维向量信号 y_2。其中，二维向量信号 y_1 为：

$$
y_1 = \begin{pmatrix} x_2(t) \\ x_3(t) \end{pmatrix}
$$

[illegible]，在这个动体仿真过程中，读示波器将显示两个物体的位移情况。

9.9.2 添加S函数的程序代码

继续上面小节的步骤。

step 1 [illegible] 模块的功能在于设置S函数实现实物的仿真，其属性如图9.115所示。

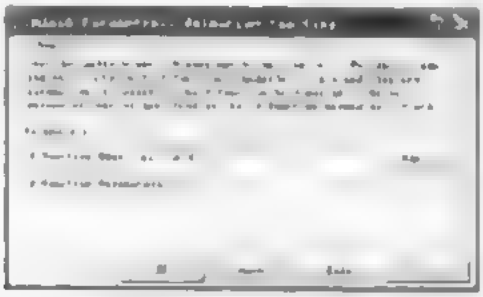


图9.115 设置“Animation function”模块的属性

step 2 [illegible] 仿真函数的代码

```
function [ sys,x0] =antsim38(t,x,u,flag,ts);
% 定义全局变量
global xSpr2 xBx12 xBx22 sim38
% 定义样本的初始的偏移数值
 [illegible];
if flag==2,
  if any(get(0,'Children')==sim38),
    [illegible],
      [illegible],
      [illegible],
      distance=u(2)-u(1);
      [illegible],
      x=[ xBx12+u(1); xSpr2/4*distance+u(1); xBx22+distance+u(1)];
      set(hndl,'XData',x);
      drawnow;
    end
  end
  sys=[];
elseif flag == 4 % 返回下一个样本
    % ns 表示样本的数目
  [illegible]
  % 返回时间变量数值
  sys = (1 + floor(ns + 1e-13*(1+ns)))*ts;
[illegible]
  % 初始化仿真系统的图形
  animinit('sim38 Animation');
  sim38 = findobj('Type','figure','Name','sim38 Animation');
  axis([ -10 20 -7 7]);
  hold on;
```

```
% 定义变量的数值
xySpr2=[ ...
        0.0        0.0
        0.4        0.0
        0.8        0.6[illegible]
        1.6       -0.6[illegible]
        [illegible]
        3.2       -0.6[illegible]
        3.6        0.0
        4.0        0.0];
[illegible]
        0.0        1.1
        0.0       -1.1
       -2.0       -1.1
       -2.0        1.1
        0.0        1.1];
[illegible]
        2.0        1.1
        [illegible]
        0.0        1.1];
% 定义运动物体的坐标
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
ySpr2=xySpr2(:,2);
[illegible]
y=[ yBx12; ySpr2; yBx22];
% 绘制两个移动物体下的坐标
plot([ -10 20],[ -1.3 -1.3],'yellow', ...
     [ -10:19:-9:20],[ -2 -1.3],'yellow','LineWidth',2);
[illegible]
set(gca,'UserData',hnd1);
[illegible]
x0=[ ];
end;
```

在上述代码中，[illegible]

[illegible]

9.9.3 添加子系统模块

继续上面小节的步骤。

Step 1 [illegible] 子系统子模块 [illegible] 'Inputs & Outputs' 模块，[illegible] 编辑器，在其中添加子系统的模块，如图 9.116 所示。

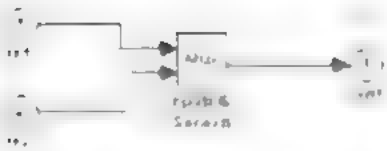


图 9.116 添加“Inputs & Sensors”子系统的模块

在本步骤中，“Inputs & Sensors”子系统的三个输入端口分别接收[illegible]个量，其中一个信号是 u，它是上级控制的输入信号；另外一个信号是一个向量信号 $x_1(t)$，而系统输出结果信号是：

$$y=\begin{bmatrix} u \\ x_1(t) \end{bmatrix}$$

step 2 添加“State estimator”子系统的模块，[illegible]“State estimator”模块，打开模块编辑器，在其中添加子系统的模块，如图 9.117 所示。

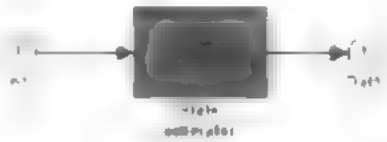


图 9.117 添加“State estimator”子系统的模块

step 3 设置“State estimator”模块的属性。在上面的子系统中，“State estimator”模块的属性对话框如图 9.118 所示。

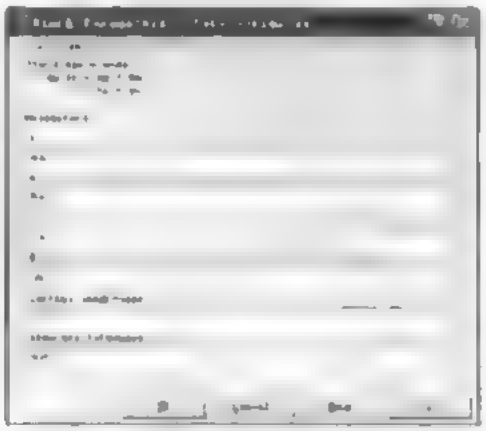


图 9.118 设置模块的属性

step 4 设置参数值。在上面的参数对话框中，ae、be、ce 和 de 分别表示状态空间系统的四个参数的矩阵参数，其具体的数值可以使用 MATLAB 的相关命令获得，结果如下：

```
>> ae
ae =

   -1.0000         0   44.8537         0
         0         0  -46.6599    1.0000
    1.0000         0  -89.5750         0

>> be
be =
```

```
         0   31.4707
    1.0000  -43.8939
            46.6539
         0   89.5750

ans =
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
>> ddr
ddr =
     0     0
     0     0
     0     0
     0     0
```

[illegible]

[illegible]

Step 5 [illegible]是进行信号选择，其属性如图9.119所示。

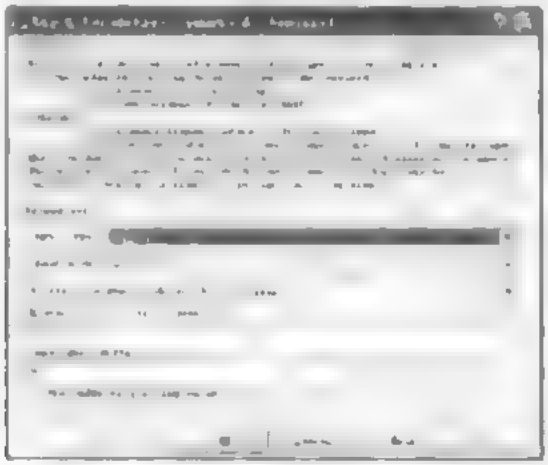


图9.119 设置信号选择的属性

[illegible]统的模块输出的$x_1(t)$、$x_2(t)$、$x_3(t)$和$x_4(t)$信号。

Step 6 [illegible]出反馈信号，其具体的子系统模块如图9.120所示。

图 9.120 添加反馈信号的模块

Step 3 [illegible]

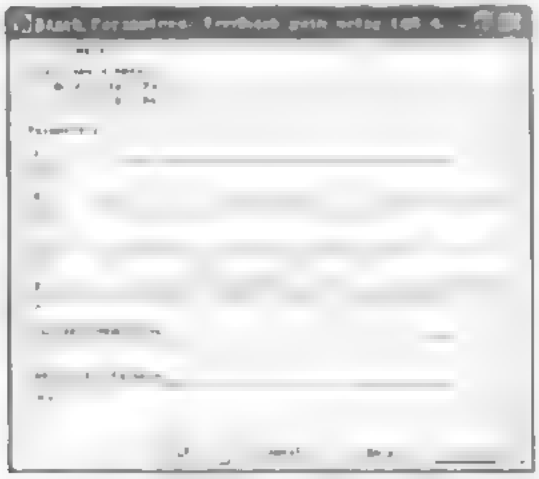


图 9.121 设置模块的属性

[illegible]

Step 4 [illegible] 设置封装系统图标的对话框如图 9.122 所示。

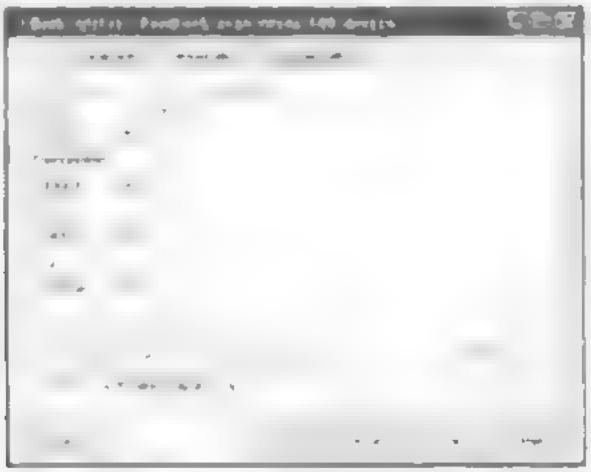


图 9.122 设置封装子系统的图标

设置封装子系统的参数对话框如图 9.123 所示。

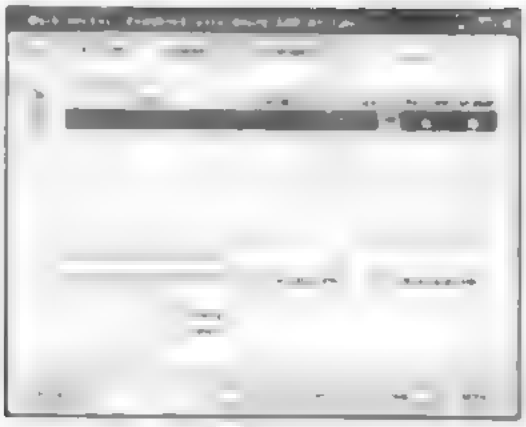


图 9.123　设置封装子系统的参数

Step 9 [illegible] "Model pre-load function" 选框中输入 "load sim38"，如图 9.124 所示。

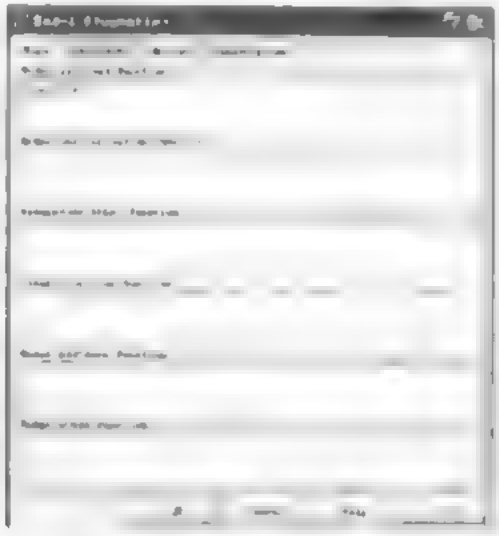


图 9.124　设置系统的加载函数

Step 10 [illegible]

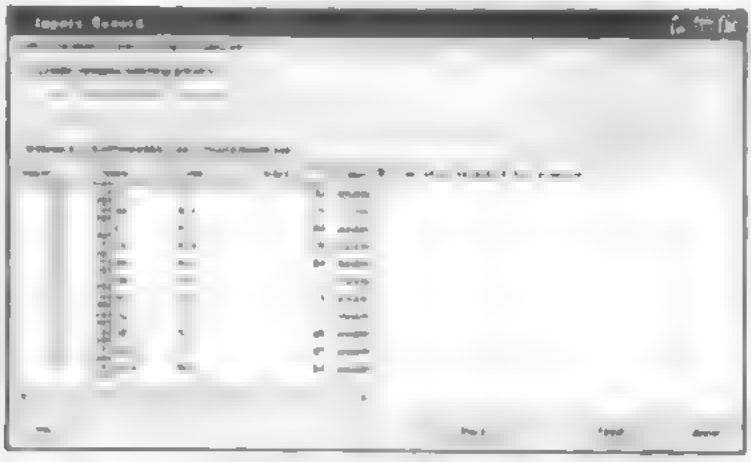


图 9.125　查看数据文件

9.9.4 运行仿真系统

[illegible]

Step 1 [illegible]系统任务，将系统任务[illegible]，然后单击“开始仿真”按钮，得到动态仿真的结果，分别出现两个动态仿真画面，如图 9.126 和图 9.127 所示。

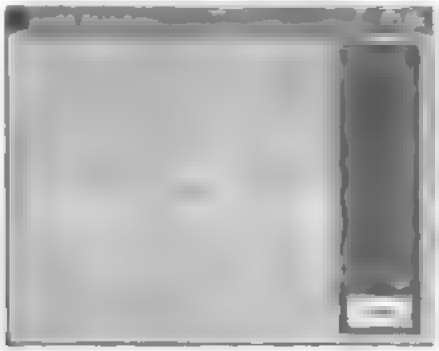


图 9.126 动态仿真画面 1

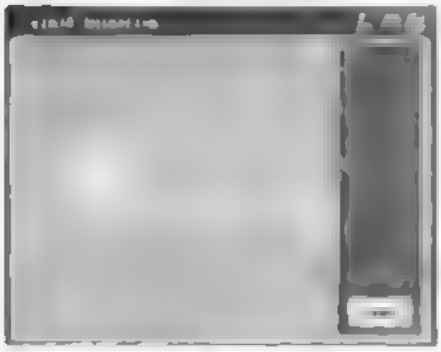


图 9.127 动态仿真画面 2

Step 2 查看二连杆的[illegible]，其中“Actual Position”示波器[illegible]，如图 9.128 所示。

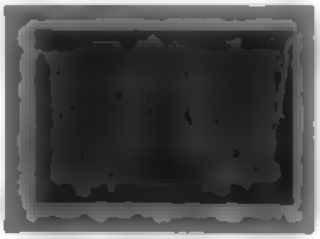


图 9.128 两个物体的真实位移图形

和通过系统仿真得到的模拟位移图形如图 9.129 所示。

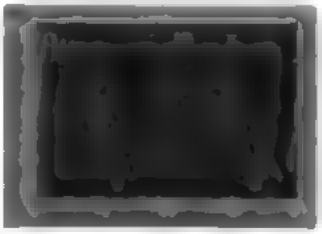


图 9.129 通过仿真得到的位移图形

[illegible]

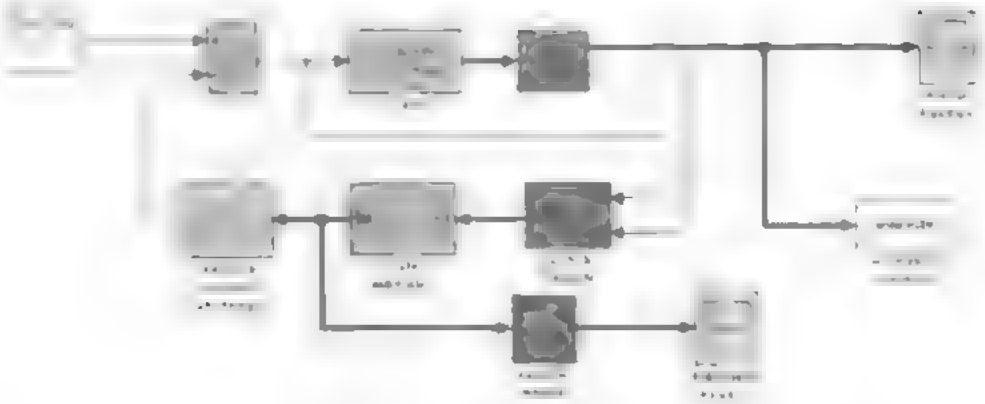


图 9.130 运行后的系统模块

[illegible]

9.10 仿真结果分析

[illegible]

9.10.1 分析 Simulink 模型的特征

[illegible]

下面以具体的实例来介绍该命令的使用方法。

例 9.12 [illegible] Anti-lock brake [illegible]

[illegible]

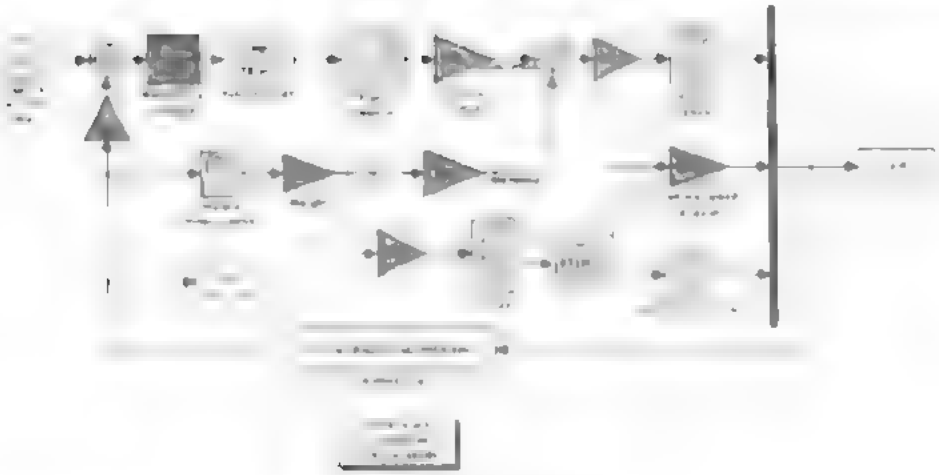


图 9.131 系统的模块

step 2 [illegible]

```
[illegible]
[illegible]
>> Sizes
Sizes =
[illegible]

>> X0
X0 =
   88.0000   70.4000         0         0         0
>> StateCell
StateCell =
    '[illegible]'
    '[illegible]'
    '[illegible]'
    '[illegible]'
    'simh2/Hydraulic Lag '
```

[illegible]

下面简要介绍上面代码的结果。

- **Sizes**：Sizes 是一个 7 维向量，对应各分量向量的含义如下。
  - **Sizes(1)**：表示状态向量中的连续分量个数。
  - **Sizes(2)**：表示状态向量中的离散分量个数。
  - **Sizes(3)**：表示输出分量个数。
  - **Sizes(4)**：表示输入分量个数。
  - **Sizes(5)**：表示系统中不连续解的个数。
  - **Sizes(6)**：表示系统中是否包含直通回路。
  - **Sizes(7)**：表示状态中不同采样速率的个数。
- **X0**：[illegible]窗口中的 X0 初始值来重新设置。

- StateCell：[illegible]模块名称。

[illegible]

9.10.2 Sim 命令

[illegible]

- [t,x,y] = sim(model);
- [illegible]
- [t,x,y1, y2, ..., yn] = sim(model,timespan,options,ut);

关于上面调用格式中的参数说明如下。

- model：[illegible]
- y：[illegible]时间变量记录。
- y1, y2, ..., yn：[illegible]
- x：[illegible]StateCell中获取。
- timespan：[illegible]
  - [ ]：[illegible]
  - T_final：标量参数，指定系统仿真的终止时间。
  - [T_start T_final]：[illegible]
- OutputTimes：任何指定输出时间记录点的向量。
- Options：[illegible]
- ut：[illegible]

9.10.3 Sim 命令实例

例 9.13 [illegible]

Step 1 [illegible]的系统模块如图 9.132 所示。

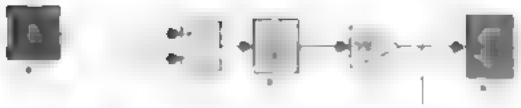


图 9.132 原始的系统模块

step 2 在 MATLAB 的命令窗口中，输入下面的命令行：

```
>> for i=1:5
[illegible]
[illegible] 'LineWidth',[illegible]
[t,x,y] = sim('Sim28',[0 10],Opts(i));
plot(t,x,'LineWidth',2),hold on
grid
end
```

[illegible]

提示

[illegible]

step 3 查看上面程序代码运行的结果，在命令窗口中，按下回车键，[illegible]，如图 9.133 所示。

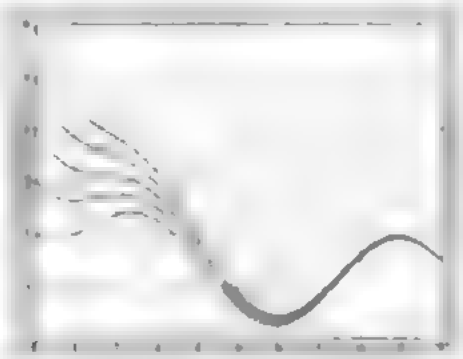


图 9.133 查看程序代码运行的结果

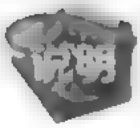
9.10.4 simset 命令

[illegible] 见调用格式如下

- options = simset(property, value, ...);
- options = simset(old_opstruct, property, value, ...);
- options = simset(old_opstruct, new_opstruct);
- simset

[illegible]

```
>> simset
          Solver: [ 'VariableStepDiscrete' |
                    'ode45' | 'ode23' | 'ode113' | 'ode15s' | 'ode23s' |
'ode23t' | 'ode23tb' | 'FixedStepDiscrete' |
                    'ode5' | 'ode4' | 'ode3' | 'ode2' | 'ode1' | 'ode14x' |
          RelTol: [ positive scalar | 1e-3] |
          AbsTol: [ positive scalar | 1e-6] |
          Refine: [ positive integer | 1] ]
         MaxStep: [ positive scalar | auto] |
         MinStep: [ [positive scalar, nonnegative integer] | auto] ]
     InitialStep: [ positive scalar | auto] |
        MaxOrder: [ 1 | 2 | 3 | 4 | 5] ]
       FixedStep: [ positive scalar | auto] |
    ExtrapolationOrder: [ 1 | 2 | 3 | 4] ]
NumberNewtonIterations: [ positive integer | 1] |
   OutputPoints: [ 'specified' | 'all' ]
OutputVariables: [ 'txy' | 'tx' | 'ty' | 'xy' | 't' | 'x' | 'y' ]
     SaveFormat: [ 'Array' | 'Structure' | 'StructureWithTime']
  MaxDataPoints: [ non-negative integer | 0] ]
     Decimation: [ positive integer | 1] ]
   InitialState: [ vector | [] ]
 FinalStateName: [ string | ''] ]
          Trace: [ comma separated list of 'minstep', 'siminfo',
'compile', 'compilestats' | '']]
   SrcWorkspace: [ 'base' | 'current' | 'parent' ]
   DstWorkspace: [ 'base' | 'current' | 'parent' ]
      ZeroCross: [ 'on' | 'off' ]
[illegible]
```



[illegible]

9.10.5 simset 命令实例

例 9.14 使用 simset 命令[illegible]。为了帮助分析过程，将沿用上面的例子。

step 1 在 MATLAB 的命令窗口中输入[illegible]

```
>> opts(1)=simset('Solver','ode15s');
[illegible]
[illegible]
hold on
[illegible]
[t,x,y] = sim('Sim28',[0 10],opts(2));
plot(t,x,'g','LineWidth',1.5)
axis([9 10 0.5 1])
[illegible]
```

Step 2 [illegible]

图 9.134 运行程序得到的结果

[illegible]

关于 simset 命令，笔者认为有必要提示读者下面的一些内容。

- [illegible]
- [illegible] 'base' | 'current' | 'parent'，[illegible] 'base' | 'current' | 'parent'。含义相同。

simset 命令[illegible]

在上一节介绍中，还曾经提示 simget 命令[illegible]。simget 的常用调用格式如下

- struct = simget(model)
- value = simget(model, property)
- value = simget(OptionStructure, property)

[illegible]

9.10.6 模型的线性化命令

[illegible]

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}$$

[illegible]

[illegible]

在Simulink中，提供两种线性分析命令，分别对连续系统和离散系统进行线性操作，其中对连续系统进行线性操作的命令如下

```
argout = linmod('sys');
argout = linmod('sys',x,u);
argout = linmod('sys', x, u, para);
argout = linmod('sys', x, u, 'v5', para);
argout = linmod('sys', x, u, 'v5', para, xpert, upert);
```

[illegible]

以使用Control Toolbox中的minreal命令来求取最小值。

9.10.7 模型的线性化实例

例9.15 演示如何使用Simulink来对非线性系统进行线性化。

Step 1 建立非线性系统的模块，在本实例中，由于只是为了演示线性化的操作方法，所以采用的非线性模块比较简单，如图9.135所示。

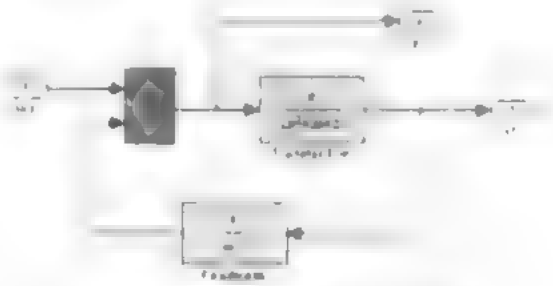


图9.135 非线性系统的模块

Step 2 对非线性模块进行线性化操作，得到线性模块的参数。在MATLAB的命令窗口中输入下面的程序代码

```
>> [A,B,C,D] = linmod('Sim40')
A =
```

```
   -1.5000   -1.0000    1.0000
    1.0000         0         0
         0    2.0000   -1.0000
B =
     1
     0
     0
C =
     0     2     -
     0     0     1
D =
     1
     1
```

step 3 根据[illegible]，在 MATLAB 的命令窗口中输入下面代码：

```
>> sys = ss(A,B,C,D)
a =
          x1     x2     x3
   x1    -1.5     -1      1
   x2      1      0      0
   x3      0      0      1
b =
          u1
   x1      1
   x2      0
   x3      0
c =
          x1   x2   x3
   y1      0    2    0
   y2      0    0    1
d =
          u1
   y1      0
   y2      0
Continuous-time model.
```

说明：[illegible]，Time-Invariant 的意思是 Simulink 中[illegible]，可以[illegible]，在对应的函数中[illegible]，[illegible]，可以查看相应的帮助文件，这里就不详细介绍了。

step 4 绘制系统的波特图，在 MATLAB 的命令窗口中输入下面代码：

```
>> bode(sys)
```

step 5 单击回车键，输入[illegible]，按"Enter"键，则会[illegible]

提示：波特图绘制[illegible]，[illegible]，关于 bode 命令的详细使用方法，请查看相应的帮助文件。

step 6 绘制系统的单位阶跃响应曲线，在命令窗口中输入下面代码：

```
>> subplot(1,2,1);step(sys);grid
```

图 9.136 系统的波特相位振幅图

step 7 [illegible]

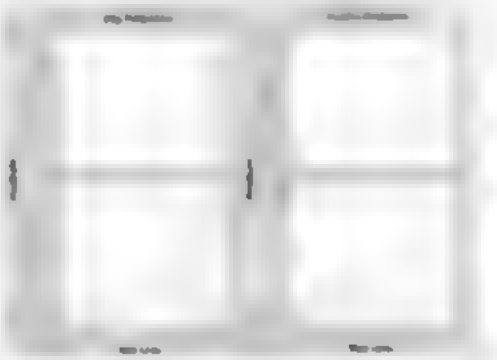


图 9.137 计算系统的响应信息

9.10.8 系统平衡点分析

[illegible] sim.trim 中，用户可通过使用 trim 命令来定义系统的稳定状态，[illegible] 性能，其具体的调用格式如下

```
[x,u,y,dx] = trim('sys',x0,u0,y0,ix,iu,iy,dx0,idx,options,t)
```

[illegible] 项数值。

例 9.16 [illegible]

step 1 [illegible] 9.138 所示。

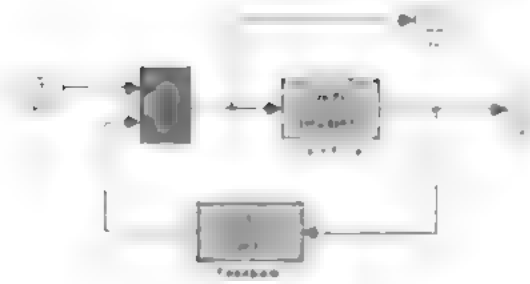


图 9.138 添加系统模块

step 2 设置命令行模式初值条件。在 MATLAB 的命令窗口中输入下面的代码

```
[illegible]
[illegible]
[illegible]
ix = [ ];
[illegible]
iy = [ 1;2] ;
```

step 3 检测稳态点的平衡性。在命令窗口中输入下面的代码

```
>> [ x,u,y,dx] = trim('Sim41',x,u,y,ix,iu,iy)
x =
     [illegible]
     0
     0
u =
     0
y =
     [illegible]
     0
dx =
     [illegible]
     [illegible]
     0
```

step 4 修改命令模式的初值条件，并检测稳态点的平衡性。在 MATLAB 的命令窗口中输入下面的代码

```
>> x = [ 0; 1; 0] ;
u = 0;
y = [ 3; 4] ;
[illegible]
iu = [ ] ;
iy = [ 1;2] ;
>> [ x,u,y,dx] = trim('Sim41',x,u,y,ix,iu,iy)
x =
   -0.0000
    [illegible]
   -0.6000
[illegible]
    1.0000
[illegible]
   -0.6000
    0.4000
```

[illegible]

[illegible]

9.11 综合实例 1：交替执行系统

作为本章实例，[illegible] Simulink [illegible] 对 Simulink 复杂模型 [illegible]

9.11.1 添加系统模块

在本[illegible]，[illegible] 所涉及到的 Simulink 组件包括 Merge 模块、[illegible] clean 信号、交替执行系统和利用符号修改[illegible]

例 9.17 [illegible]

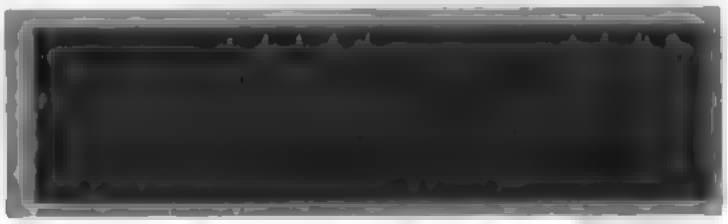


图 9.139 交替执行系统的图形

[illegible]

Step 1 [illegible] 系统模块如图 9.140 所示

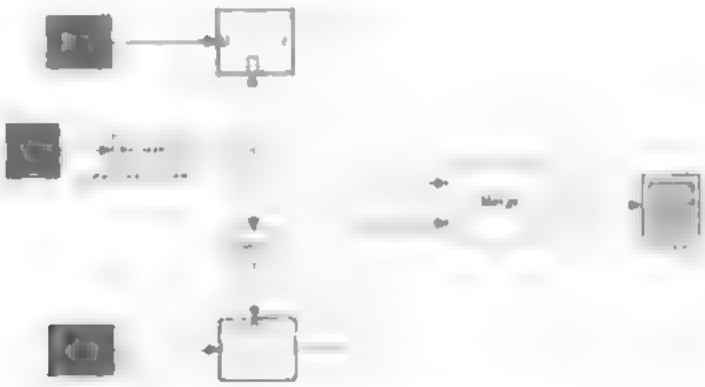


图 9.140 添加的系统模块

step 2 系统模块功能[illegible]，[illegible]系统模块比较复杂，[illegible]建议读者[illegible]系统模块的功能，在本步骤中将简要介绍模块的功能。

- **输入信号**：[illegible]"Repeating Sequence"模块[illegible]模块[illegible]模块[illegible]模块[illegible]个输入信号，其中[illegible]模块[illegible]
- **条件执行系统**：[illegible]"Enabled"子系统[illegible]，两个子系统的[illegible]"Discrete-Time Integrator"模块产生[illegible]，系统将以5s为周期，数值为[illegible]，当控制信号输入数值为1时，系统将会触发上面的子系统，当控制信号输入为0时，系统将会触发下面的子系统。
- **Merge 模块**：此[illegible]系统输出的信号作为Merge模块的输入信号，[illegible]信号，在 Scope 模块中显示。

9.11.2 设置系统模块的属性

继续上面小节的步骤。

step 1 设置"Sine Wave"模块的属性。[illegible]，[illegible]单个系统[illegible]规，因此需要设置对应的属性，如图9.141所示。

图 9.141 设置正弦波的属性

[illegible]，将[illegible]设置为1，如果设置为[illegible]，将[illegible]设置为1，得到周期为2s的正弦波图形。

step 2 设置"Repeating Sequence"模块的属性，[illegible]输入信号，需要设置"Repeating Sequence"模块产生信号的属性，如图9.142所示。

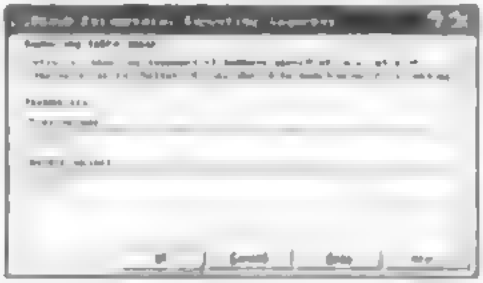


图 9.142 设置 Repeating Sequence 信号的属性

Step 3 [illegible] 参数"Output values"指定对应信号时间内的信号振幅。

[illegible]

[illegible]，输入的值参数[illegible]，[illegible]折线波形，幅值为2。

Step 4 [illegible]过封装的子系统模块，其对应的子系统如图9.143所示。

图9.143 模块对应的子系统模块

Step 5 [illegible]模块[illegible]，其[illegible]模块[illegible]，且[illegible]如下

```
/*
* [illegible]
* [illegible]
#define S_FUNCTION_NAME  sfun_tstart
#include "simstruc.h"
/* [illegible] */
static void mdlInitializeSizes(SimStruct *S)
{
    /* [illegible] */
    if (ssGetNumSFcnParams(S) != ssGetSFcnParamsCount(S)) {
        /* Return if number of expected != number of actual parameters */
        [illegible]
    }
    /* [illegible] */
    if (!ssSetNumOutputPorts(S, 1)) return;
    [illegible]
    [illegible]
    [illegible]
    [illegible]
                    SS_OPTION_WORKS_WITH_CODE_REUSE |
                    [illegible]

/* [illegible] */
static void mdlInitializeSampleTimes(SimStruct *S)
```

```
{
    ssSetSampleTime(S, 0, CONTINUOUS_SAMPLE_TIME);
    ssSetOffsetTime(S, 0, FIXED_IN_MINOR_STEP_OFFSET);
    ssSetModelReferenceSampleTimeDefaultInheritance();
}
/* Function: mdlStart =======================================================
#define MDL_START
static void mdlStart(SimStruct *S)
{
    real_T *y = (real_T *)ssGetOutputPortSignal(S,0);
    *y = ssGetTStart(S);

/* Function: mdlOutputs =====================================================
static void mdlOutputs(SimStruct *S, int_T tid)
{
    /* start time set in mdlStart, never changes */


/*=============================*
 * Required S-function trailer *
 *=============================*/
#define MATLAB_MEX_FILE    /* Is this file being compiled as a MEX-file? */
#include "simulink.c"      /* MEX-file interface mechanism */
#else
#include "cg_sfun.h"       /* Code generation registration function */
#endif
```

由于上面的代码是经过编译、连接的，此过程本书中没有涉及，这里就不详细分析了。之所以列出上面的代码，是为了方便读者理解模块产生的原理。由于该模块是经过封装的模块，可以在封装属性对话框中设置模块的初始化属性，如图 9.144 所示。

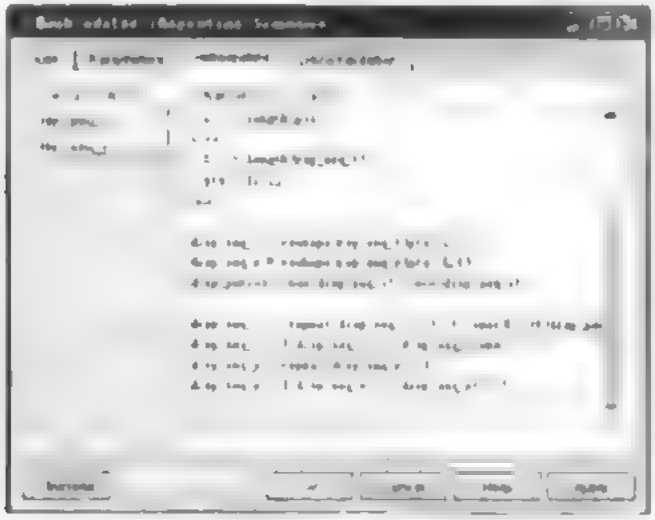


图 9.144 模块的初始化属性

step 6 设置模块的属性。单击模块，选择[illegible]，设置[illegible]，如图9.145所示。

图9.145 设置控制信号模块的属性

在一个[illegible]模块，将"[illegible]"参数设置为"[illegible]"，这样，该模块将会产生一个用户指定的波形可编辑重复波形。

step 7 设置"Data Type Conversion"模块的属性。[illegible]，因此需要将输入信号转换为"Boolean"数据类型，选择该模块属性如图9.146所示。

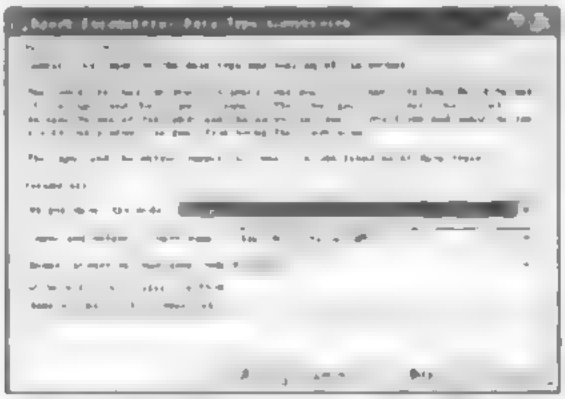


图9.146 设置"Data Type Conversion"模块的属性

step 8 设置[illegible]属性，[illegible]"[illegible] Parameters"字段，[illegible]"[illegible] to Parameters"[illegible]，选择其中[illegible]，在[illegible]"Implement [illegible] logic as Boolean data"选项，如图9.147所示。

提示

[illegible]模块[illegible]，需要在[illegible]中对[illegible]属性设置

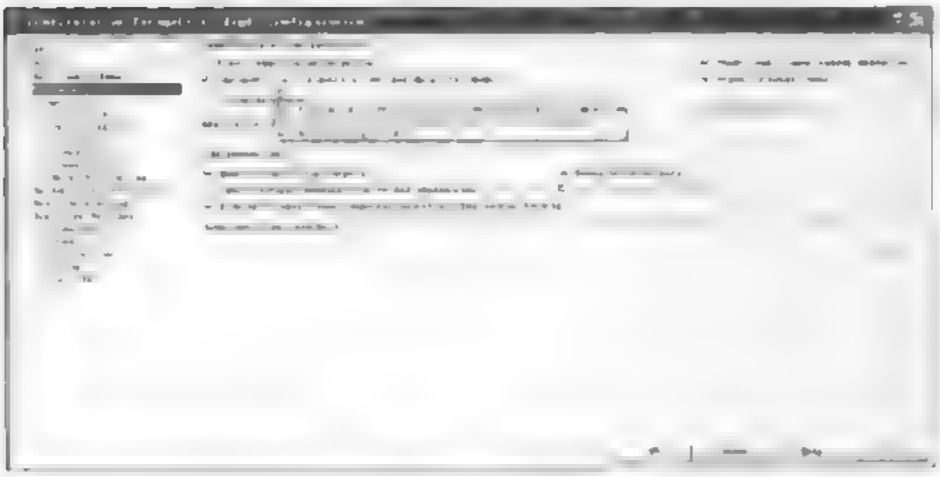


图 9.147 设置 Simulink 的属性

9.11.3 添加"Enabled"子系统

接续上面小节的步骤

Step 1 设置[illegible]子系统的模块，在上[illegible]系统模块，打开模块编辑器，在其[illegible]子系统的模块。如图 9.148 所示。

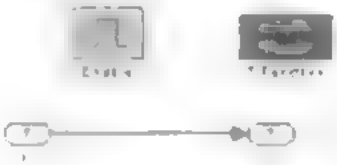


图 9.148 添加子系统模块

Step 2 设置"Enable"模块的属性，在上一步系统中的"Enable"模块，[illegible]的属性对话框，设置的属性如图 9.149 所示。

图 9.149 设置"Enable"模块的属性

Step 3 设置[illegible]模块属性，在上一步系统中的[illegible]模块，打开对应的属性对话框，设置属性如图 9.150 所示。

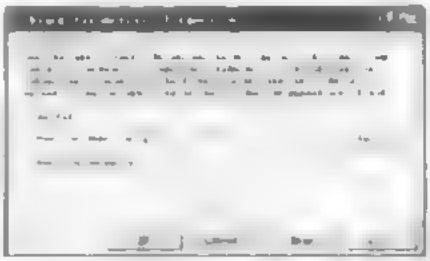


图 9.150 设置模块属性

step 4 添加S函数的程序代码。单击上面对话框中的"Edit"按钮，查看S函数"mergefcn"的具体代码如下：

```
function [ sys,x0,str,ts] = mergefcn(t,x,u,flag)
% S-Function for Simulink merge demonstration.
switch flag,
    case 0
    [ sys,x0,str,ts]=mdlInitializeSizes;
  case 2
    sys=mdlUpdate(t,x,u);
  case 9
    sys = mdlTerminate;
  case { 1, 3, 4 }
    sys=[];
  otherwise
    error([ 'Unhandled flag = ',num2str(flag)]);
end
%=============================================================================%
function [ sys,x0,str,ts] = mdlInitializeSizes()
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 0;
sizes.NumInputs      = 0;
sizes.DirFeedthrough = 1;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
str = [];
x0  = [];
ts  = [-1 0];   % inherited sample time
%=============================================================================%
function sys = mdlUpdate(t,x,u)
root = get_param(bdroot,'Handle');
subs = find_system(root,'Tag','MergeExample');
parent = get_param(get_param(gcbh,'Parent'),'Handle');
notme = subs(find(subs ~= parent));
me = subs(find(subs == parent));
if ~strcmp(get_param(me,'BackgroundColor'),'green')
  set_param(me,'BackgroundColor','green');
  drawnow
  set_param(notme,'backgroundcolor','white')
end
shortpause
sys = [];
function sys = mdlTerminate
root = get_param(bdroot,'Handle');
subs = find_system(root,'Tag','MergeExample');
set_param(subs,'Backgroundcolor','white')
sys = [];
function shortpause
pause(0.1)
```

上面程序代码主要功能是实现两个信号的融合，读者可以参考前面章节中关于S函数的内容来分析上面的代码，这里主要分析上面的mdlUpdate函数，其程序代码如下：

```
function sys = mdlUpdate(t,x,u)
root = get_param(bdroot,'Handle');
[illegible]
parent = get_param(get_param(gcbh,'Parent'),'Handle');
notme = subs(find(subs ~= parent));
me = subs(find(subs == parent));
if ~strcmp(get_param(me,'BackgroundColor'),'green')
  set_param(me,'BackgroundColor','green');
  drawnow
  set_param(notme,'backgroundcolor','white')
end
shortpause
sys = [];
```

[illegible]

```
set_param(me,'BackgroundColor','green');
```

其他程序代码和上面代码的功能类似，读者可以自行分析。

Step 5 设置"NOT"模块的属性。[illegible]的属性，如图9.151所示。

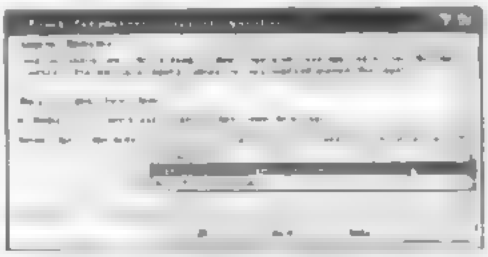


图9.151 设置"NOT"模块的属性

[illegible]

Step 6 设置"Merge"模块的属性。[illegible]模块的属性，如图9.152所示。

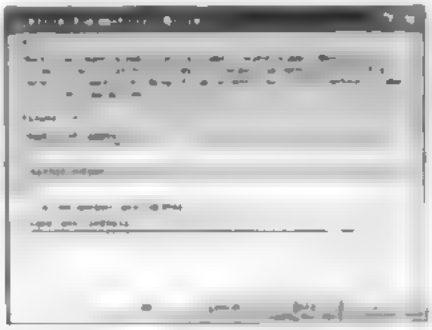


图9.152 设置"Merge"模块的属性

9.11.4 运行仿真系统

[illegible]

Step 1 [illegible]

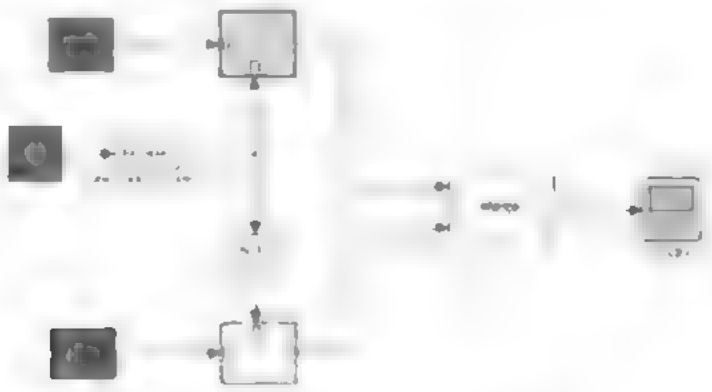


图 9.153 执行上面的子系统

[illegible]

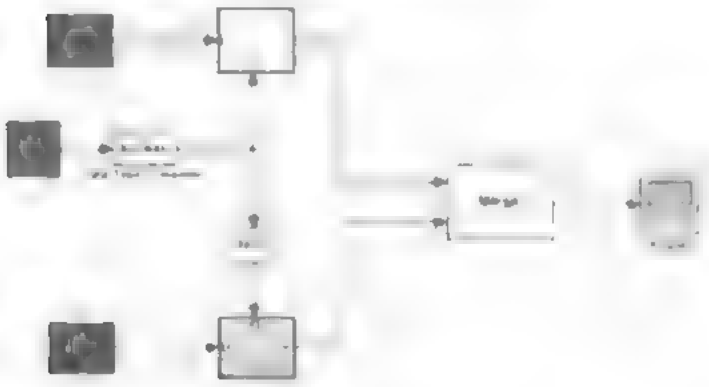


图 9.154 执行下面的子系统

Step 2 查看仿真结果。同时，得到的仿真[illegible]

图 9.155 仿真结果图形

9.12 综合实例 2：雷达轨迹分析

[illegible]

需要通过Simulink中的封装子系统、S函数或算法各种单元功能，使模型变得比较复杂，下面将分小节详细介绍。

9.12.1 系统模块简介

例 9.18 使用Simulink来创建雷达飞行跟踪系统，并将雷达测量数据与飞机实际轨迹数据的图形，其中雷达获取的飞机运动轨迹和模拟的运动轨迹如图中所示。

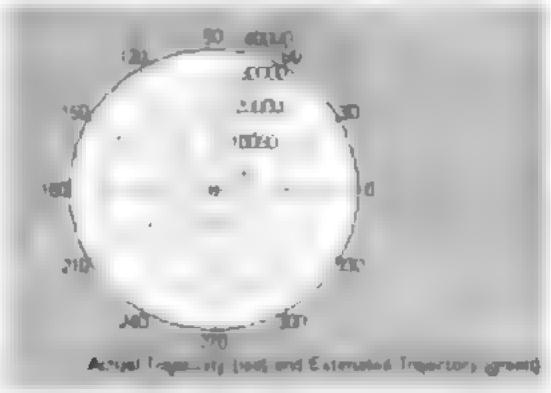


图 9.156 雷达获取的飞机轨迹

同时，为了分析系统模拟的精度，需要分析雷达获取的轨迹和模拟轨迹的偏差情况，需要绘制偏差的图形，如图9.157所示。

最后，观察飞机在不同方向运动轨迹的对比情况，如图9.158所示。

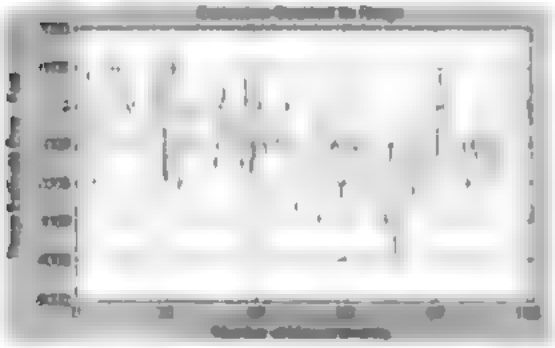


图 9.157 实际轨迹和模拟轨迹的误差曲线

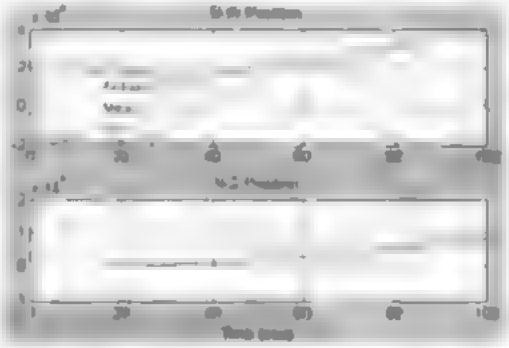


图 9.158 不同方向轨迹的对比情况

9.12.2 添加系统模块

下面将详细分析上面系统的创建过程。

step 1 创建系统模块，在本节开始所创建的系统模块的基础上进行修改。

step 2 分析系统模块功能。由于这个系统比较复杂，在详细介绍系统各个模块参数设置和功能之前，首先介绍该系统模块的整体功能。

- 求解飞机在第一个采样时刻轨迹分析。在"Carrier-to-Noise"模块中创建系统模块，其主要功能是求解飞机在第一个采样时刻的轨迹数值。首先通过"Random aircraft motion"模块产生的随机数值作为飞机运动的初始信号，接收信号分别通过"CMSS-Axis

[illegible]

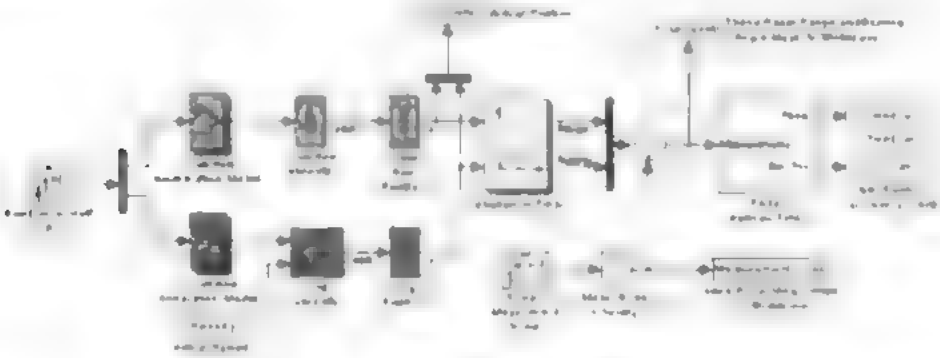


图 9.159　创建系统模块

- [illegible]
- [illegible]
- [illegible]

Step 3 设置系统模块的属性。由于系统模块繁多，[illegible]

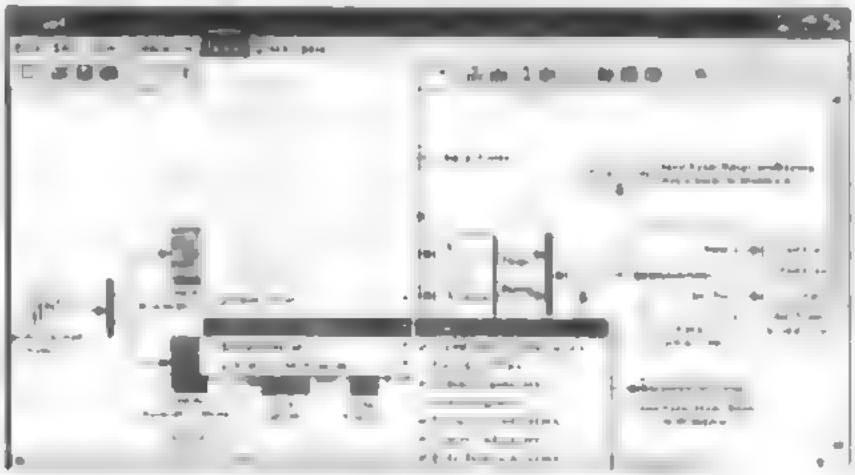


图 9.160　设置模块的采样属性

Step 4 设置系统的解决器。[illegible]

的名称"[illegible]"选项卡中的"type"选项中选择"[illegible]"，在"solver"选项中选择"ode5(Dormand-Prince)"选项，如图 9.161 所示。

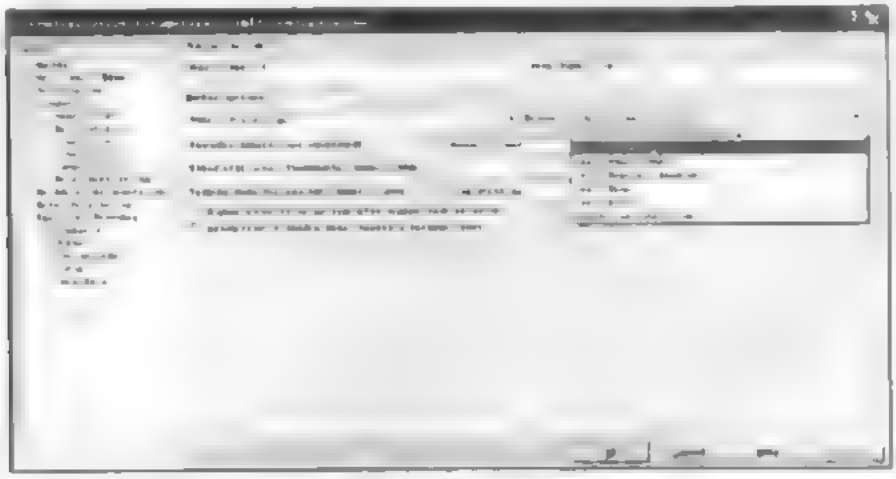


图 9.161 设置系统的解法器

[illegible]

step 3 设置"Random aircraft motion"模块的属性。单击工具栏中的"Random aircraft motion"模块，打开模块的属性对话框，在其中设置其属性，如图 9.162 所示。

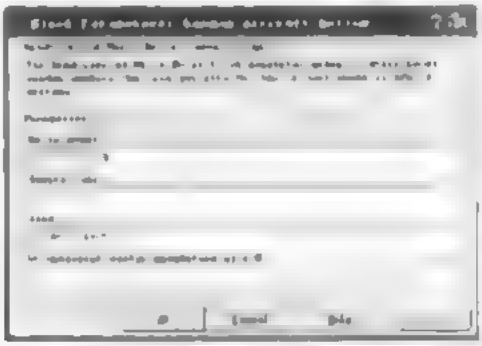


图 9.162 设置"Random aircraft motion"模块的属性

在系统中"Random aircraft motion"模块的功能是产生[illegible]数值，[illegible]模块。选择该模块，单击鼠标右键，在弹出的快捷菜单中选择"Link Options"→"Go to Library Block"命令，打开 Simulink 库浏览器，[illegible]该模块在模块库中位置，如图 9.163 所示。

如要编辑该模块，必须先选择该模块，单击鼠标右键，在弹出的快捷菜单中选择"Link Options"→"Break Link"命令，断开该模块和模块库的链接，[illegible]，在弹出的快捷菜单中选择"Edit Mask"选项，[illegible]该模块的编辑器。

例如，如果希望修改该模块的初始值，[illegible]"Initial Value"选项卡，在其中设置模块的初始数值。如图 9.164 所示。

图 9.163　Simulink 中的模块库

图 9.164　设置模块的初始数值

9.12.3　添加“Cross-Axis Acceleration Model”子系统

继续上面小节的步骤，

step 1　添加“Cross-Axis Acceleration Model”子系统的模块。与子系统“Cross-Axis Acceleration Model”子系统，打开模块编辑框。添加以下的子系统，如图 9.165 所示。

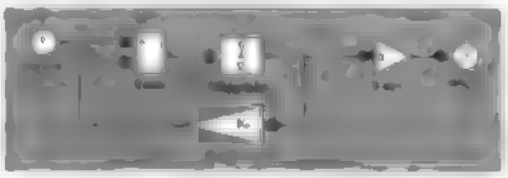


图 9.165　添加子系统模块

step 2　分析上面子系统模块功能，根据 Simulink 的基础知识，上面的子系统模块相当于下面的微分方程组

$$\begin{cases} I - Kx = x' \\ \alpha = x \end{cases}$$

其中，I 表示由负载引起的惯性冲击，K 表示弹簧的刚度系数，x 表示弹簧的压缩或拉伸的变量，α 是模块输出的横轴加速度。

上面系统其他的复杂的子系统内部模块内容，"Thrust Axial Acceleration Mode" 子系统模块的建模方法在结构上完全相同，只是参数在数值上不同。

9.12.4 添加"Cartesian to Polar"子系统

继续上面小节的步骤。

Step 1 添加"Cartesian to Polar"子系统的模块，双击工具栏中的"Cartesian to Polar"子系统，打开模块编辑器，添加新的子系统，如图 9.166 所示。

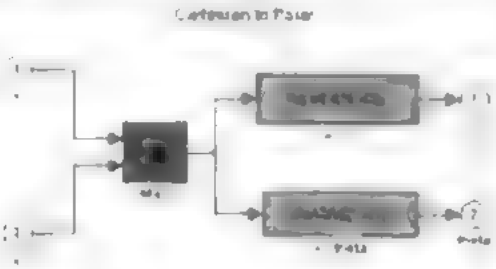


图 9.166 添加"Cartesian to Polar"子系统的模块

在上面的系统模块中，hypot 和 atan2 都是 MATLAB 中内置的数学函数，分别将输入的 x 和 y 坐标数值转换为极坐标下的 r 和 θ，其具体的计算公式如下：

$$\begin{cases} r=\sqrt{x^2+y^2} \\ \theta=\alpha\tan(\frac{y}{x}) \end{cases}$$

Step 2 封装"Cartesian to Polar"子系统的模块，选择"Cartesian to Polar"子系统模块，按住鼠标右键，在弹出的快捷菜单中选择"Mask Subsystem"选项，打开模块编辑器，选择"Icon"选项卡，在其中设置封装子系统的图标，如图 9.167 所示。

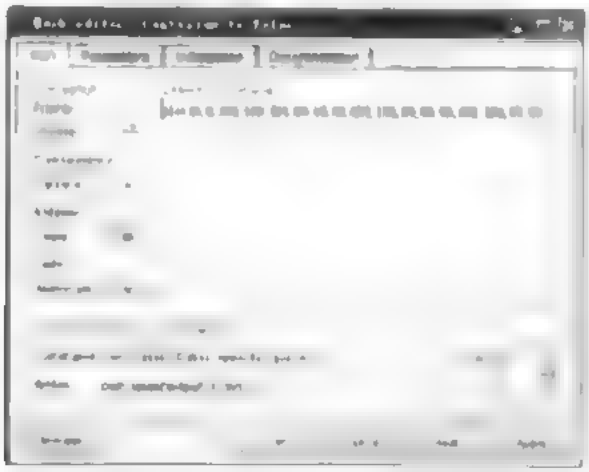


图 9.167 设置封装子系统的图标

Step 3 设置"Radar Measurement Noise"模块的属性，双击系统中的"Radar Measurement Noise"模块，打开属性对话框，在其中设置对应的属性，如图 9.168 所示。

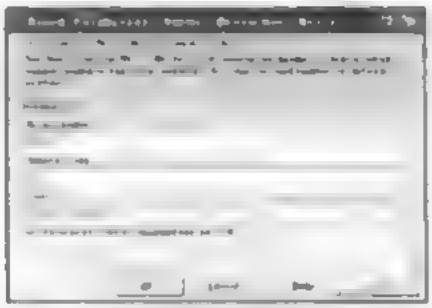


图 9.168　设置“Radar Measurment Noise”模块的属性

[illegible]

step 4 双击“Meas. Noise Intensity”模块的属性，在弹出的“Meas. Noise Intensity”模块[illegible]，在其中设置相应的属性，如图 9.169 所示。

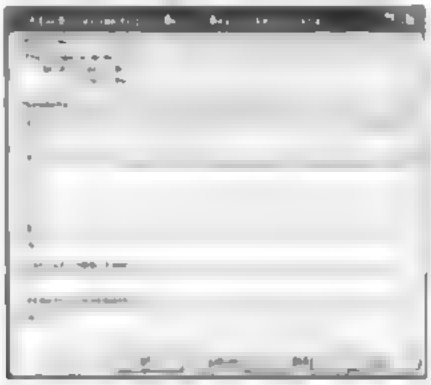


图 9.169　设置模块的参数值

在上述操作过程中，双击模块设置参数值K时，其实是在重新定义参数K的值，而[illegible]步骤的操作方便，有必要对该模块进行封装。

step 5 选择“Meas. Noise Intensity”模块，选择“Meas. Noise Intensity”子系统模块，然后单击鼠标右键，在弹出的快捷菜单中选择“Mask Subsystem”选项，打开模块编辑器，点击“Parameters”选项卡，设置封装子系统的参数，如图 9.170 所示。

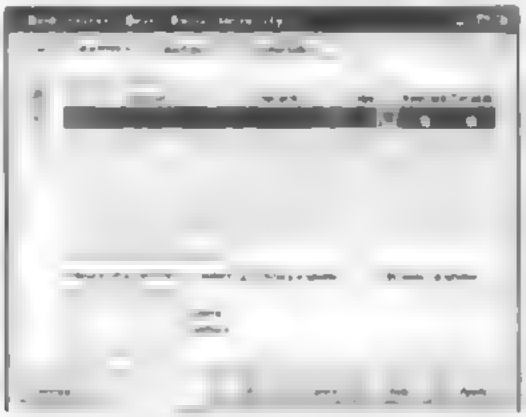


图 9.170　设置封装模块的参数

对子系统封装模块的参数值，源下篇幅，这里就不一一赘述了，[illegible]

图 9.171 完成后的封装子系统

9.12.5 添加"Radar Kalman Filter"子系统

继续上面小节的步骤。

step 1 新建"Radar Kalman Filter"子系统的模块，双击子系统的"Radar Kalman Filter"模块，打开模块编辑器，在其中添加子系统的模块，如图 9.172 所示。

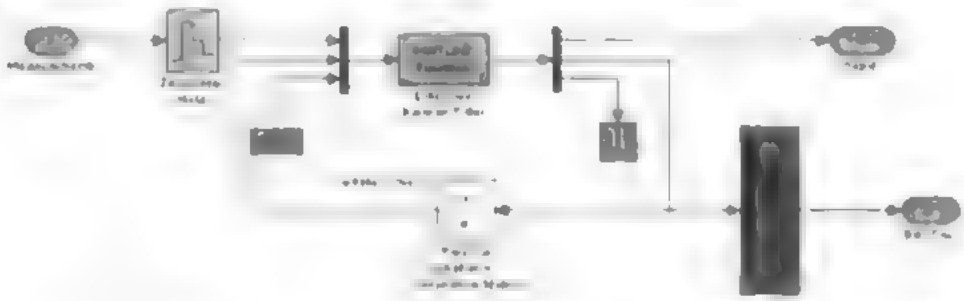


图 9.172 添加"Radar Kalman Filter"子系统的模块

step 2 设置"Zero-Order Hold"模块的属性。该子系统主要功能是将来自模型的数据经过Kalman Filter处理，在在"Zero-Order Hold"模块上右键的属性对话框，在其中设置其属性，如图 9.173 所示。

图 9.173 设置"Zero-Order Hold"模块的属性

在 Simulink 中，"Zero-Order Hold"模块[illegible]

step 3 设置"Extended Kalman Filter"模块的属性。双击子系统中的"Extended Kalman Filter"模块，打开属性对话框，设置该模块的属性，如图 9.174 所示。

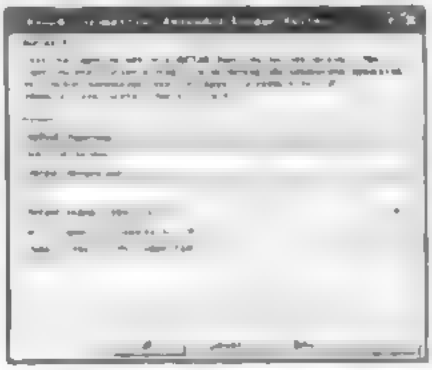


图 9.174　设置模块的属性

Step 4 单击按钮，在弹出的"MATLAB function"选项中输入"aero_extkalman"，这是M文件的名称，其具体的代码如下

```
function funcout = aero_extkalman pinput)
   [illegible] the derivation of the "Extended Kalman Filter" Brochure for
%  the equations.
% Initialization
meas = pinput(1:2);
xhatPrev = pinput(3:6);
PPrev = pinput(7:22);          % Covariance matrix (zeros assumes perfect
estimate)
deltat = pinput(23);
xhat = xhatPrev(:);            %  Estimate
P    = reshape(PPrev,4,4);
   [illegible] update time = deltat [illegible] inherited from workspace where it was
defined by raddat.
% 1. Compute Phi, Q, and R
Phi = [ 1 deltat 0 0; 0 1 0 0 ; 0 0 1 deltat; 0 0 0 1];
Q =  diag([ 0 .005 0 .005]);
R =  diag([ 300^2 0.001^2]);
% 2. Propagate the covariance matrix:
P = Phi*P*Phi' + Q;
% 3. Propagate the track estimate::
xhat = Phi*xhat;
% 4 a). Compute observation estimates:
Rangehat = sqrt(xhat(1)^2+xhat(3)^2);
Bearinghat = atan2(xhat(3),xhat(1));
   4 b). Compute observation vector y and linearized measurement matrix
M
yhat =    [Rangehat
          Bearinghat];
M = [ cos(Bearinghat)          0 sin(Bearinghat)          0
   -sin(Bearinghat)/Rangehat 0 cos(Bearinghat)/Rangehat 0 ];
% 4 c).  Compute residual (Estimation Error)
residual = meas - yhat;
% 5. Compute Kalman Gain:
W = P*M' [illegible] R);
% 6. Update estimate
xhat = xhat + W*residual;
% 7. Update Covariance Matrix
P = (eye(4)-W*M)*P*(eye(4)-W*M)' + W*R*W';
% Output [illegible]
```

```
funcout = [ residual;xhat;P(:);deltat];
```

在上面的程序代码中，依次按照步骤计算Extended Kalman Filter中的转换工作，得到的结果就是将极坐标数值转换为Kalman Filter处理后的数据。

添加程序代码

延续上面小节的步骤。

step 1 编写系统参数的M文件，其具体代码如下：

```
%RADDAT 是运行仿真系统所需要的基础参数值
g = 32.2;      % 加速度（相对于重力加速度）
tauc = 5;      % 横轴加速的时间
tauT = 4;      % 推力轴加速的时间
Speed = 400;  % 在y方向上的初始速度
deltat = 1;
```

上面的数据都是仿真系统的基础参数数据，将上面的代码保存为“aero_raddat.m”文件，在后面的步骤中将在运行仿真系统之前加载这些数据。

step 2 编写绘制参数图形的M文件，其具体代码如下：

```
%RADPLOT
%delt = 0.1;   %  仿真的样本时间
%deltat = 5;   %  雷达更新的时间
% Post Processing of the Data for Plotting:
% 在极坐标条件下绘制飞机轨迹
pos = [10 40 500 300];
h_1 = figure(1);
set(h_1,'pos',pos);
polar(PolarCoords(:,2) - Measurement_noise(:,2), ...
      PolarCoords(:,1) - Measurement_noise(:,1),'r')
hold on
rangehat = sqrt(X_hat(:,1).^2+X_hat(:,3).^2);
bearinghat = atan2(X_hat(:,3),X_hat(:,1));
polar(bearinghat,rangehat,'g')
text(-35000,-50000,'Actual Trajectory (red) and Estimated Trajectory
(green)')
% 创建新的图形窗口，绘制误差曲线
h_2 = figure(2);
set(h_2,'pos',[pos(1)+500 pos(2) pos(3:4)]);
plot(residual(:,1)); grid;set(gca,'xlim',[0 length(residual)]);
xlabel('Number of Measurements');
ylabel('Range Estimate Error - Feet')
title('Estimation Residual for Range')
% 创建新的图形窗口，绘制对比情况
h_3 = figure(3);
set(h_3,'pos',[pos(1) pos(2)+350 pos(3:4)]);
XYMeas = [PolarCoords(:,1).*cos(PolarCoords(:,2)), ...
          PolarCoords(:,1).*sin(PolarCoords(:,2))];
t_full = [0:0.1:100]';
t_hat = [0:deltat:100]';
subplot(211)
plot(t_full,XYCoords(:,2),'r');
```

```
grid on;hold on
plot(t_full,XYMeas(:,2),'g');
[illegible]
[illegible]
legend('Actual','Meas','Est',3);
hold off
subplot(212)
plot(t_full,XYCoords(:,1),'r');
[illegible]
[illegible]
[illegible]
xlabel('Time (sec)');
[illegible]
hold off
```

[illegible]

Step 3 [illegible]，在该选项卡对应的选框中加载上面的程序代码，如图9.175所示。

图9.175 设置系统模块的属性

9.12.7 运行仿真系统

继续上面小节的步骤。

Step 1 [illegible]，如图9.176所示。

Step 7 [illegible]

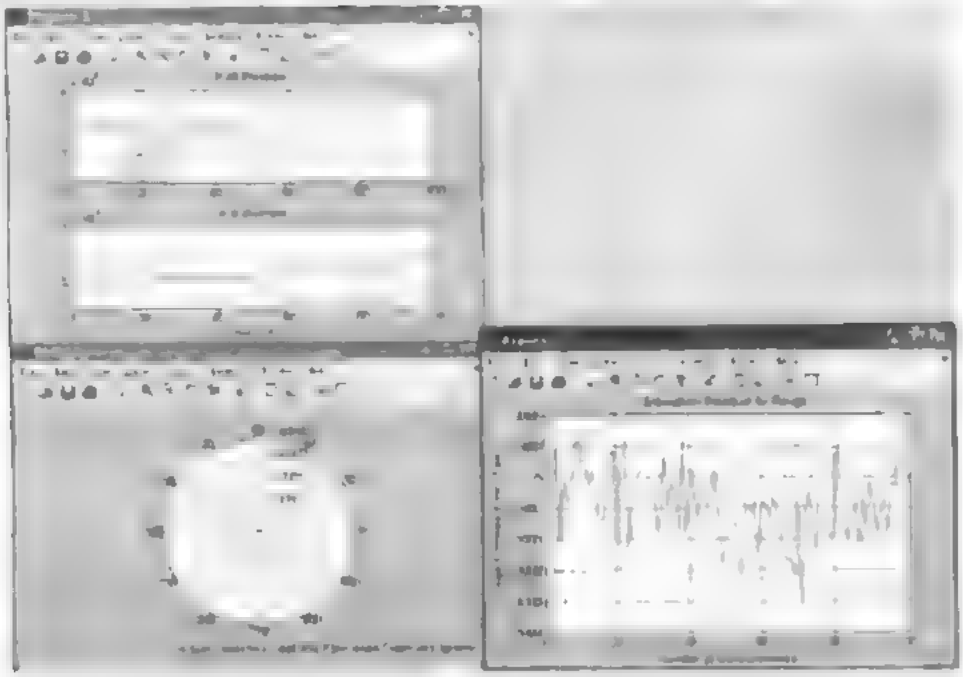


图 9.176 得到的仿真结果

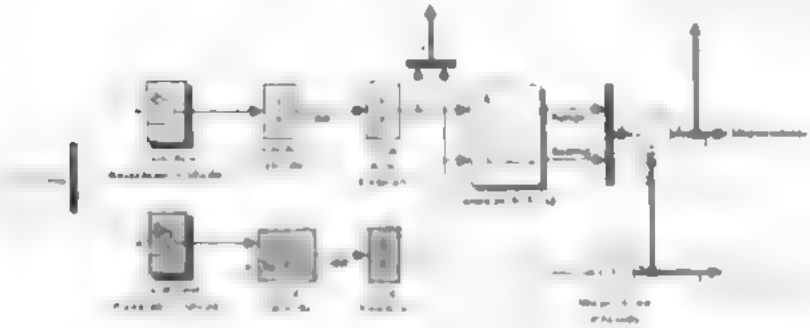


图 9.177 运行的系统模块

9.13 小结

[illegible]

◆ ◆ ◆

第10章 句柄图形

本章概述

- 句柄图形体系
- 图形对象的操作
- 坐标轴对象
- 图形句柄的操作
- 高层绘图命令

[illegible]

MATLAB中的句柄图形绘制各种复杂图形。

[illegible]

10.1 句柄图形体系

[illegible]

句柄图形。

[illegible]

10.1.1 图形对象

[illegible]

图形，还有坐标轴、线条、曲面、文本和其他对象。

[illegible]

10.1.2 句柄对象

在MATLAB中，每一个对象名称[illegible]，[illegible]，[illegible]建一个对象[illegible]MATLAB[illegible]建。一个[illegible]，[illegible]句柄[illegible]。其他对象句柄是MATLAB高精度的浮点值。

[illegible]

[illegible]

10.1.3 句柄图形的结构

在MATLAB中，所有图形[illegible]，[illegible]句柄图形[illegible]。[illegible]

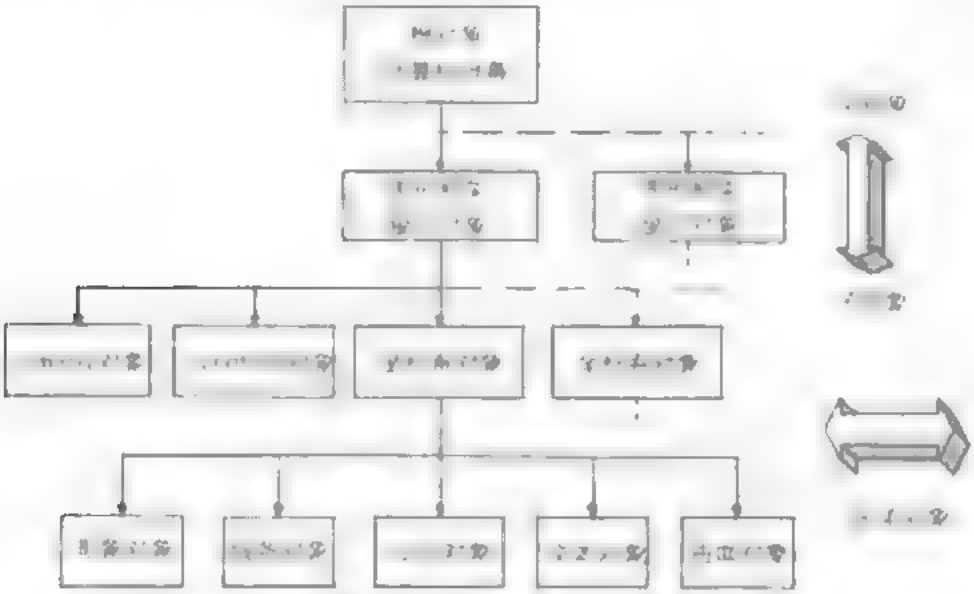


图 10.1 句柄图形的结构

[illegible]

10.1.4 图形对象的属性

[illegible]

[illegible]"Position"，甚至"pos"来调用。

[illegible]

[illegible]

[illegible]

10.2 图形句柄的操作

[illegible]

10.2.1 创建图形对象

[illegible]

[illegible]

表 10.1 列出了 MATLAB7.0 中所有图形对象的创建函数。

表 10.1 MATLAB 中的图形对象创建函数

函数	说明
Axes	创建图形的坐标轴对象
Figure	创建或者显示图形窗口对象
[illegible]	[illegible]
[illegible]	[illegible]
Line	由[illegible]坐标数据的直线段构成的线条
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
Text	创建位于坐标轴系统内的字符串对象
Uicontrol	创建用户界面的控件
[illegible]	[illegible]

10.2.2 创建图形对象实例

例 10.1 [illegible]

step 1 [illegible]

```
[illegible]
[illegible]
>> [X,Y]=meshgrid(x,y);
[illegible]
>>ah=axes('color',[0.8 0.8 0.8],'xtick',[-4 -3 -2 -1 0 1 2 3 4],'ytick',
[-4 -3 -2 -1 0 1 2 3 4]);
[illegible]
[illegible]
'edgecolor','k','marker','o','markerfacecolor',[0.5 1 .85]);
```

step 2 [illegible]

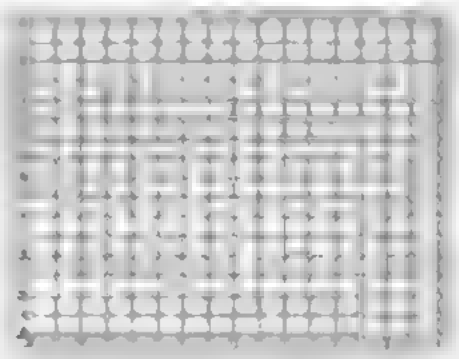


图 10.2 创建图形对象的实例

[illegible]

[illegible]

[illegible]

Step 3 [illegible]

```
view(3)
```

Step 4 [illegible]

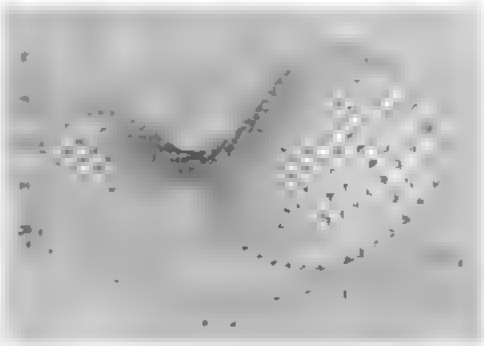


图 10.3 修改视角后的图形

Step 5 [illegible]

```
>> text1 = text(...
   'Position',[-36.6 -45.59 186.6],...
   [illegible]
```

Step 6 [illegible]

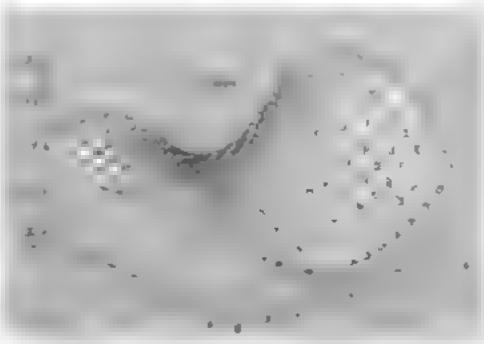


图 10.4 添加了文本对象的图形

[illegible]

```
>> text1 = text(-36.6,-45.59,186.6,'Figure1');
```

该命令和上面步骤中使用的命令是等效的。

[illegible]

10.2.3 访问图形对象的句柄

[illegible]

在 MATLAB 中，获取图形对象的句柄有下面几种常见的方法。

- [illegible]

  ```
  >>H Line1=plot(x,y);
  >> text1 = text(-36.6,-45.59,186.6,'Figure1');
  ```

  [illegible]

- [illegible]

  [illegible]

- [illegible]，MATLAB 提供[illegible]

  - gcf　返回当前图形窗口（CurrentFigure）的句柄
  - [illegible]
  - [illegible]

- [illegible]

  ```
  [illegible]
  >>set(gca,'Tag','A1')
  ```

10.2.4 访问图形句柄实例

[illegible]

例 10.2　[illegible]对象句柄，修改文字注释的位置。

Step 1　在 MATLAB 命令窗口输入下面[illegible]

```
>> t=0:0.01:2*pi; y=cos(t);
>> plot(t,y) grid on
```

Step 2　查看绘图结果。输入下面[illegible]键，[illegible]

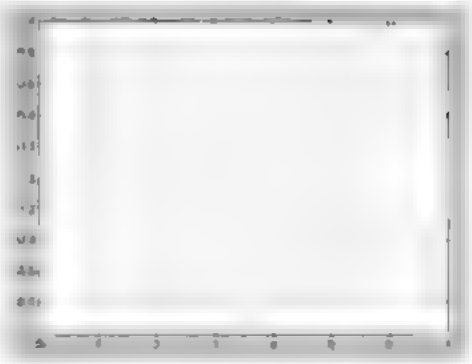


图 10.5　绘制的基础图形

step 3 [illegible]

```
>> text(5,0.8,'\fontsize{16}cos(t)','Tag','A1');
```

step 4 [illegible]

图 10.6　添加文字注释

step 5 [illegible]

```
>> H=findobj(0,'Tag','A1');
>> set(H,'position',[3 0.8])
```

step 6 [illegible]

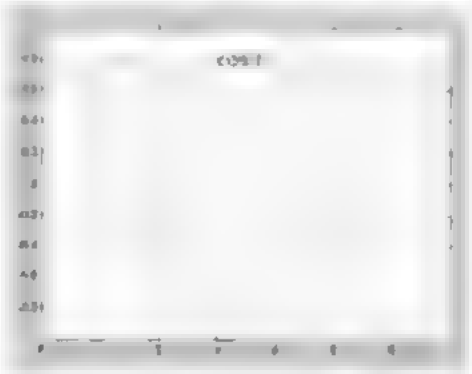


图 10.7　修改文本的位置属性

step 7 [illegible]

象类型，在命令窗口中输入下面的内容

```
>> H_figure=get(gca,'children');
[illegible]
T =
    'text'
    'line'
```

[illegible]，[illegible]函数，获取相应的句柄和句柄类型，其中T是元胞数组类型。

10.2.5 使用句柄操作图形对象

[illegible]函数[illegible]，MATLAB[illegible]，使用[illegible]，[illegible]，[illegible]，[illegible]

[illegible]函数[illegible]对象中，其调用格式如下

```
new_handle = copyobj(h,p)
```

[illegible]parent[illegible]轴。[illegible]对象[illegible]，MATLAB则会将子对象一起复制过去。

例 10.3 [illegible]

step 1 [illegible]

```
>> h = surf(peaks);
>> colormap hot
```

step 2 [illegible]

```
[illegible]
[illegible]
[illegible]
```

[illegible]peaks[illegible]子对象，程序得到的图形如图 10.8 所示。

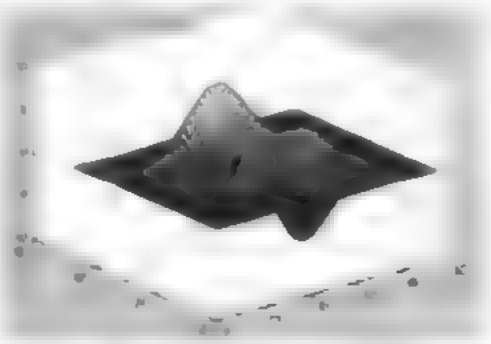


图 10.8 旋转的图形对象结果

step 3 [illegible]

```
>> figure        % Create a new figure
>>axes          % Create an axes object in the figure
>>new_handle = copyobj(h,gcf);
```

Step 4 [illegible]

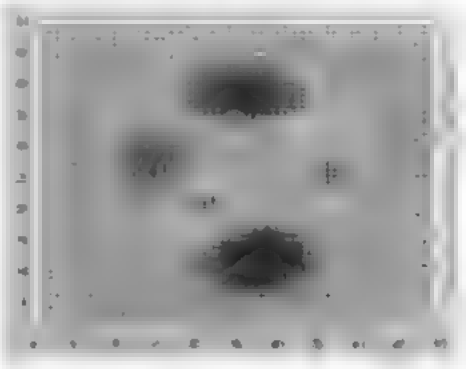


图 10.9 复制的图形对象

说明

从上面的操作结果来可以看出，只要使用 copyobj 命令复制了图形对象的属性，它就会与原始的 surf 的默认，保持同，[illegible]

Step 5 [illegible]

```
>> colormap cool
[illegible]
>>grid on
```

Step 6 [illegible]

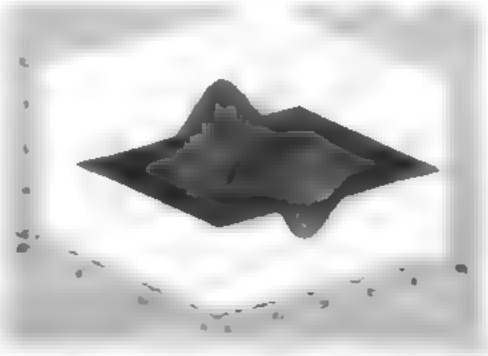


图 10.10 修改后的图形对象

除了使用句柄函数外，MATLAB 还提供了 plot 命令来直接绘制图形对象，使用户能够[illegible]，而且不必要担心其他对象[illegible]。且对象的句柄仍使用 plot 命令来绘制对象。

例 10.4 [illegible]

Step 1 在 MATLAB 命令行窗口中输入下面的命令

```
[illegible]
>> plot(t,y)
>>grid on
```

[illegible]

step 2 [illegible]

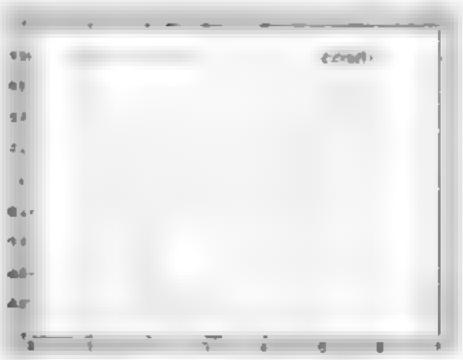


图 10.11 原始图形

step 3 [illegible]

```
[illegible]
>> delete(H)
```

step 4 [illegible]

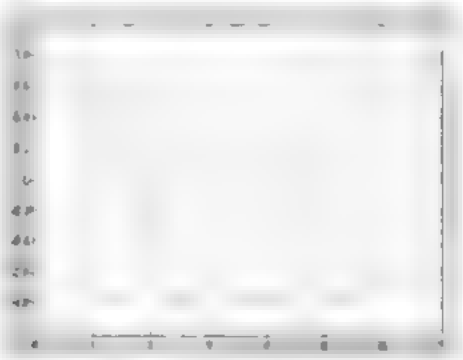


图 10.12 删除文本对象

10.3 图形对象的操作

[illegible]

10.3.1 set 命令

[illegible]

◆ [illegible]

```
H_GC=GraphicCommand(......,PN,PV)
```

其中，GraphicCommand 代表的是 MATLAB 中可用的绘图命令，可以是高层命令，也可以是底层命令。(PN，PV) 是属性名、属性值构成的属性对，属性对的数目没有任何的限制。例如，下面的命令行就可以在创建对象的时候设置相应的属性：

```
>> plot(x,y,'r*','LineWidth'...
2,'MarkerSize',20)
```

◆ 通过函数 set 设置图形对象的属性。在 MATLAB 中，set 函数的常用调用格式如下：

```
set(H,'PropertyName',PropertyValue,...)  设置H句柄对象的对应属性的属性值
set(H,'PropertyName')                    显示H句柄对象PropertyName的全部属性值
```

例如，可以通过 set 函数将图形的 y 坐标轴移到图形的右侧，对应的代码如下：

```
set(gca,'YAxisLocation','right')
```

set 命令实例

例 10.5 绘制 peaks 函数的三维图形，并通过 set 函数查看各种属性。

step 1 绘制三维图形。在 MATLAB 的命令窗口中输入下面的代码：

```
>> H_figure=mesh(peaks(40));
```

上面的命令行可以得到peaks函数的曲面图，同时将图形句柄保存在变量H_figure中。为了节省篇幅，这里就不给出具体的三维图形了。

step 2 通过 set 函数查看用户可以设置的图形属性，得到的结果如下：

```
>>set(H_figure)
ans =
                AlphaData: {}
          AlphaDataMapping: {3x1 cell}
                   CData: {}
            CDataMapping: {2x1 cell}
               EdgeAlpha: {2x1 cell}
               EdgeColor: {3x1 cell}
               EraseMode: {4x1 cell}
               FaceAlpha: {3x1 cell}
               FaceColor: {4x1 cell}
               LineStyle: {5x1 cell}
               LineWidth: {}
                  Marker: {14x1 cell}
          MarkerEdgeColor: {3x1 cell}
          MarkerFaceColor: {3x1 cell}
              MarkerSize: {}
               MeshStyle: {3x1 cell}
                   XData: {}
                   YData: {}
                   ZData: {}
            FaceLighting: {4x1 cell}
            EdgeLighting: {4x1 cell}
```

```
    BackFaceLighting: (3x1 cell)
     AmbientStrength: []
.......................................//限于篇幅,省略了部分属性数值
          BusyAction: (2x1 cell)
    HandleVisibility: (3x1 cell)
             HitTest: (2x1 cell)
       Interruptible: (2x1 cell)
            Selected: (2x1 cell)
  SelectionHighlight: (2x1 cell)
                 Tag: []
       UIContextMenu: []
[illegible]
[illegible]
[illegible]
         DisplayName: []
           XDataMode: (2x1 cell)
[illegible]
[illegible]
[illegible]
           CDataMode: (2x1 cell)
         CDataSource: []
         ZDataSource: []
```

[illegible]

step 3 [illegible] 'Marker' [illegible]

```
[illegible] 'Marker',
{ + | o | * | . | x | square | diamond | v | ^ | > | < | pentagram |
hexagram | [none] }
```

[illegible] 'Marker' [illegible] 设置是形标记的属性。

step 4 [illegible]

```
[illegible] 'Marker', [illegible]
```

step 5 [illegible] 'hover' [illegible]

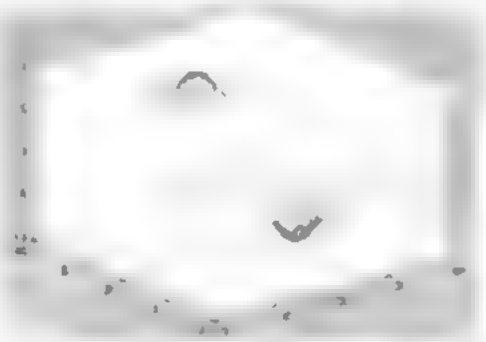


图 10.13 设置数据标记后的图形

使用结构体设置属性

除了上面的方法之外，在MATLAB中还可以使用结构体数组来直接设置图表对象的属性。例如，可以定义一个结构体来设置图表对象的属性：

```
props.FaceColor= 'texture';
props.EdgeColor = 'none';
props.FaceLighting = 'phong';
```

然后通过set函数选用上面的结构体props，设置图形对象的属性，相应的代码如下：

```
set(gcf,props)
```

相当于对当前图形窗口设置结构体props所定义的对应的属性。下面将介绍几个综合设置图形对象属性的例子。

例 10.6 在MATLAB中绘制克莱因瓶（Klein bottle）。

step 1 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的"File" ⇨ "New" ⇨ "M-file"命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
n = 12;
a = input('Enter the diameter of the small tube:');  % 小径的直径
c = input('Enter the diameter of the bulb:');        % 大径的直径
t1 = pi/4 : pi/n : 5*pi/4;       % 沿着管子的参数
t2 = 5*pi/4 : pi/n : 9*pi/4;     % 环绕管子的角度
u  = pi/2 : pi/n : 5*pi/2;
[X,Z1] = meshgrid(t1,u);
[Y,Z2] = meshgrid(t2,u);

% 绘制图形的把手
len = sqrt(sin(X).^2 + cos(2*X).^2);
x1 = c*ones(size(X)).*(cos(X).*sin(X) ...
    - 0.5*ones(size(X))+a*sin(Z1).*sin(X)./len);
y1 = a*c*cos(Z1).*ones(size(X));
z1 = ones(size(X)).*cos(X) + a*c*sin(Z1).*cos(2*X)./len;
handleHndl=surf(x1,y1,z1,X);
set(handleHndl,'EdgeColor',[.5 .5 .5]);
hold on;

% 绘制径
r = sin(Y) .* cos(Y) - (a + 1/2) * ones(size(Y));
x2 = c * sin(Z2) .* r;
y2 = - c * cos(Z2) .* r;
z2 = ones(size(Y)) .* cos(Y);
bulbHndl=surf(x2,y2,z2,Y);
set(bulbHndl,'EdgeColor',[.5 .5 .5])
colormap(hsv);
axis vis3d
view(-37,30);
axis off
light('Position',[2 -4 5])
light
hold off
```

[illegible]

step 2 [illegible]

```
>> klein
Enter the diameter of the small tube:0.3
Enter the diameter of the bulb:0.7
```

step 3 [illegible]

图 10.14 绘制得到的克莱因瓶

step 4 [illegible]

```
>> klein
Enter the diameter of the small tube:0.1
Enter the diameter of the bulb:0.9
```

step 5 [illegible]

图 10.15 修改曲面参数绘制图形

[illegible]

例 10.7 在 MATLAB 中绘制二维的曲线曲形。

step 1 [illegible] 下面的代码

```
%加载系统自带的数据文件
load topo;
%绘制球体
[x,y,z] = sphere(50);
cla reset
axis square off
%设置结构体props的属性
props.AmbientStrength = 0.1;
props.DiffuseStrength = 1;
props.SpecularColorReflectance = .5;
props.SpecularExponent = 20;
props.SpecularStrength = 1;
props.FaceColor= 'texture';
props.EdgeColor = 'none';
props.FaceLighting = 'phong';
%定义图形的数据
props.Cdata = topo;
[illegible]
%设置光照的属性
light('position',[-1 0 1]);
light('position',[-1.5 0.5 -0.5], 'color', [.6 .2 .2]);
%设置图形的视角
view(3)
```

step 2 在[illegible]文件夹中保存为文件 'three_globe.m'，[illegible]，输入 'three_globe'，得到的结果如图 10.16 所示。

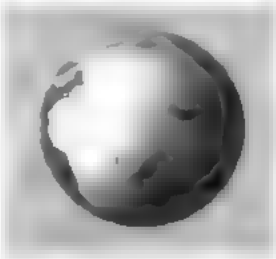


图 10.16 程序绘制得到的图形

在[illegible]"load topo"，函数[illegible]MATLAB 的[illegible]

在[illegible]结构体，通过结构体[illegible]属性。

10.3.4 查询图形对象的属性

在 MATLAB 中，[illegible]get 函数查询[illegible]对象的属性值，其常用的调用格式如：

- get(h)：获取 h 句柄对象所有属性的当前值
- [illegible]

[illegible]

例 10.8 [illegible] get 函数获取各种对象的属性。

step 1 [illegible] 在 MATLAB 命令窗口中输入下面代码

```
>>pos = [0.1 0.1 0.5 0.5];
```

[illegible] 程序代码中建了 4 个图形对象，[illegible] MATLAB 会为各个对象分配默认属性来创建对象。到时，在创建坐标对象之前，[illegible] 图形窗口或者坐标轴对象，MATLAB 会使用默认属性来创建这个对象。

step 2 查看创建结果，输入程序代码[illegible]，按"Enter"键，得到[illegible]，如下。

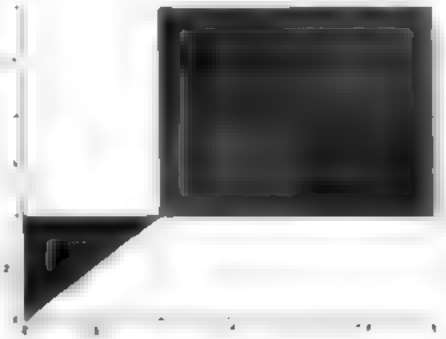


图 10.17 创建的图形对象

step 3 查看创建对象的所有属性值。在 MATLAB 命令窗口中输入"get(gca)"，得到当前绘制的图形对象的所有属性值，结果如下

```
>> get(gca)
    ActivePositionProperty = outerposition
    ALim = [ 0.1 10]
    ALimMode = auto
    AmbientLightColor = [ 1 1 1]
    Box = off
    CameraPosition = [ 1.5 1.5 [illegible]
    CameraPositionMode = auto
    CameraTarget = [ 1.5 1.5 0.5]
    CameraTargetMode = auto
    CameraUpVector = [ 0 1 0]
    CameraUpVectorMode = auto
    CameraViewAngle = [ 6.6086]
    CameraViewAngleMode = auto
    [illegible]
    [illegible]Mode = auto
    [illegible] = [ 1 1 1]
    CurrentPoint = [  (2 by 3) double array]
    ColorOrder = [  (7 by 3) double array]
    ..... ..... .....//限于篇幅，省略了部分属性数值
    Interruptible = on
    Parent = [ 1]
    Selected = off
    SelectionHighlight = on
    Tag =
```

```
    Type = axes
    UIContextMenu = []
    UserData = []
    Visible = on
```

在上面的属性数值列表中，许多属性选项都列出了具体的数值，例如"CameraPosition = [1.5 1.5 9.16025]"，但是，也有一些属性列表只是列出了数值的维度，例如"ColorOrder = [(7 by 3) double array]"。如果需要了解对应属性的数值，则需要调用相应的格式，查看具体数值。

step 4 查看该图形对象的"ColorOrder"属性值。在MATLAB的命令窗口中输入"get(gca,'ColorOrder')"，得到的结果如下：

```
>> get(gca,'ColorOrder')
ans =
         0         0    1.0000
         0    0.5000         0
    1.0000         0         0
         0    0.7500    0.7500
    0.7500         0    0.7500
    0.7500    0.7500         0
    0.2500    0.2500    0.2500
```

step 4 定义结构体，查询结构体中指代的图形对象属性。在MATLAB的命令窗口中输入代码：

```
>> props = { 'HandleVisibility', 'Interruptible';
'SelectionHighlight', 'Type'};
>>output = get(get(gca,'Children'),props);
```

step 6 查看查询结果。在MATLAB的命令窗口中输入"output"，按"Enter"键，查看属性查询结果，得到的结果如下：

```
>> output
output =
    'on'    'on'    'on'    'line'
    'on'    'on'    'on'    'text'
    'on'    'on'    'on'    'surface'
    'on'    'on'    'on'    'patch'
```

查看图形对象的默认属性

在MATLAB中，如果需要了解图形对象的所有系统默认属性，可以通过命令代码get(0,'factory')获得，得到的结果如下：

```
                factoryFigureAlphamap: [ 1x64 double]
            factoryFigureBackingStore: 'on'
              factoryFigureBusyAction: 'queue'
          factoryFigureButtonDownFcn: ''
               factoryFigureClipping: 'on'
      factoryFigureCloseRequestFcn: 'closereq'
                  factoryFigureColor: [ 0 0 0]
               factoryFigureColormap: [ 64x3 double]
             factoryFigureCreateFcn: ''
```

```
                factoryFigureDeleteFcn: ''
                [illegible]: 'on'
.....//限于篇幅,省略了部分属性数值
                [illegible]: 'on'
                [illegible]: 'on'
            factoryRootRecursionLimit: 2.1475e+009
      factoryRootScreenPixelsPerInch: 96
                [illegible]: 'on'
                [illegible]: 'off'
                     factoryRootTag: ''
                [illegible]:
                factoryRootVisible: 'on'
```

从上面的结果可以看出，系统默认的属性名称的标识符为 'factory<root object>'，而属性值为默认属性数值。

如果希望一次获取某个特定属性的数值，可以使用下面的语句获得

```
>> get(0,'factoryFigurePaperPosition')
ans =
    0.2500    2.5000    8.0000    6.0000
```

对于所有不同对象的属性，MATLAB 系统都会有一个默认的多种属性值

在 MATLAB 中，每一个对象都有系统默认属性值，在一个理论意义上，这些属性值是可以定义的，这些属性值称为对象的属性默认值。在一个图形对象的默认属性值上面，将会发现 MATLAB 定义了属性值的字符。

在 MATLAB 中运行一个名称要求能够修改属性值时，MATLAB 会从一个对象开始，在内部表现为一个字符，然后在系统默认的表现为一个表示属性默认值的属性值。换句话，定义了默认的图形对象（root 对象），MATLAB 搜索的顺序如图所示。例如，在 line 对象定义 line 对象的默认值，MATLAB 会在 root 对象中定义 line 对象的默认值。如果在 root 对象中定义 line 对象的默认值，MATLAB 会在 root 对象中定义 line 对象的默认值。如果在 root 对象中定义 line 对象的默认值，MATLAB 会在 root 对象中定义 line 对象的默认值。

如果在多种层次结构下对象的属性值，用户与对象有关的，对象的默认值将被使用，与对象默认值是 MATLAB 中最先找到的默认值

对象属性的表达方式，由于对象的默认值，可以重新定义一个新的对象类型，对于，如果建立的图形对象，设置属性的默认值的语句，在一个特定的命令，一般情况下，需要建立一个 'DefaultLine' 的属性，然后建立对象类型，除了对象属性的子对象，例如，line 对象的属性，设置一个对象的 'LineWidth' 的参数默认属性值，例如，可以使用下面的语句来实现

```
set(gcf,'DefaultLineLineWidth',2.5)
```

10.3.6 设置不同级别的属性

例 10.9 在 MATLAB 中设置图形对象的一个默认属性，并观察效果。

step 1 在 MATLAB 的命令窗口中输入下面的语句

```
>> set(0,'DefaultSurfaceMarker','o');
>>h=surface(sphere(30));
```

```
>>view(3);
>> grid
```

step 2 在命令窗口中输入如下命令，按"Enter"键，得到如图 10.18 所示。

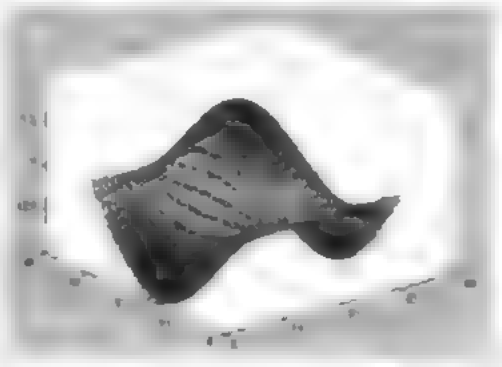


图 10.18 得到的初始图形

step 3 重新设置标记类型的默认值。在 MATLAB 的命令窗口中输入下面的代码：

```
>> set(gcf,'DefaultSurfaceMarker','*');
>> set(h,'Marker','default')
```

step 4 查看修改结果。输入绘图代码，按"Enter"键，得到如图 10.19 所示。

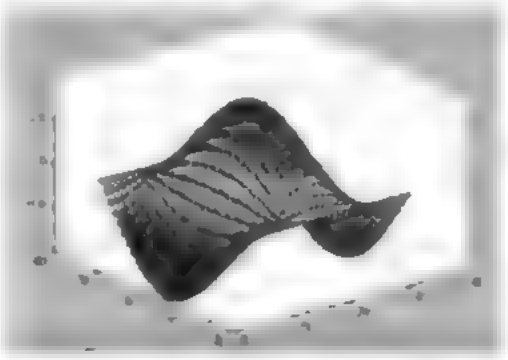


图 10.19 修改图形标记的默认值

step 5 打开新的绘图窗口，绘制一个球面。在命令窗口中输入下面的代码：

```
>> figure
>> h=surface(sphere(40));
>>view(3);
>> grid
```

step 6 查看绘图结果。输入绘图代码，按"Enter"键，得到如图 10.20 所示。

从上面的例子可以看出，get 函数和 set 函数分别用来获取和设置图形对象的属性值。在第 2 个步骤中，设置默认属性值，属性值的格式为"属性名，属性值"。get 函数和 set 函数的属性值只适用于当前绘图窗口中的图形。因此，在步骤 5 开始绘制新的图形对象时，使用的是步骤 1 中设置的默认属性值。

图 10.20 曲面绘制图形

[illegible]

10.3.7 设置图形对象的默认属性

在 MATLAB 中，除[illegible]对象的默认属性外，也可以根据需要单独设置[illegible]的默认属性。如果需要删除设置过的默认属性，可以使用下面的程序语句：

```
set(0,'DefaultSurfaceMarker','remove');
```

在输入[illegible]后，再查看[illegible]的默认属性，保证[illegible]

```
>> get(0,'DefaultSurfaceMarker')
ans =
none
```

从上面的程序语句中可以看出，在删除用户设置的默认属性后，MATLAB 系统恢复系统设定的默认属性 none，表明已经删除用户设定的默认属性。除了可以删除用户设定的默认属性之外，也可以不使用用户定义的默认属性，后面之[illegible]，在 MATLAB 的命令窗口中输入下面的程序语句：

```
[illegible],'DefaultSurfaceMarker','[illegible]');
[illegible]
[illegible]
[illegible],'Marker','[illegible]')
>> grid
```

当输入[illegible]后，按"Enter"键，[illegible]

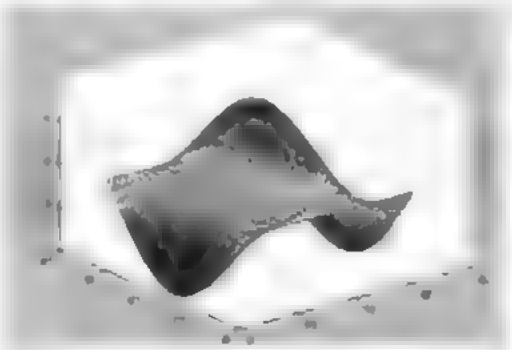


图 10.21 程序得到的结果

[illegible]

例 10.10 [illegible]

step 1 [illegible]

```
>> whitebg('w');
[illegible]
'-|:|--|-.',...
'DefaultAxesYGrid','on','DefaultlineLineWidth',2);
>>z=sphere(50);
>>plot(1:51,z(3:6,:))
```

step 2 [illegible]

图 10.22 绘制得到的图形

[illegible]

step 3 [illegible]

```
>> figure
[illegible]
>>y2=cos(t);
[illegible]
```

step 4 [illegible]

[illegible]

图 10.23 绘制新的图形

10.4 高层绘图命令

[illegible]

10.4.1 NextPlot 属性

[illegible]

Figure 对象的“NextPlot”属性如下。

- **Add**：[illegible]
- **Replacechildren**：[illegible]
- **Replace**：[illegible]

Axes 对象的“NextPlot”属性如下。

- **Add**：[illegible]
- **Replacechildren**：[illegible]
- **Replace**：[illegible]

在 MATLAB 中，Figure 的“NextPlot”默认属性是“Add”，Axes 的“NextPlot”默认属性是“Replace”。[illegible]属性。

10.4.2 Newplot 命令

[illegible]

检查并设置当前图形窗的“NextPlot”属性

- 如果结果是ReplaceChildren，则清除图形中全部的子对象；
- 如果结果是Replace，则清除图形中全部的子对象，并将图的对象属性设置为系统的默认值；
- 如果结果是Add，则保留当前图形窗口中所有子对象和所有属性不变。

检查并设置当前图形轴的“NextPlot”属性：

- 如果结果是ReplaceChildren，则清除图形轴中全部的子对象，
- 如果结果是Replace，则清除图形轴中全部的子对象，并将轴的对象属性设置为系统的默认值；
- 如果结果是Add，则保留当前图形轴中所有子对象和所有属性不变。

高层绘图文件的构成

前面已经列出MATLAB所有相关的底层绘图命令，而且已经向读者介绍过。MATLAB的高层绘图函数都建立在底层绘图命令之上，下面分析MATLAB的mesh.m文件，说明mesh命令是如何建立在surface命令之上的。

例 10.11 分析高层绘图命令 mesh 的文件构成。

在 MATLAB 的默认路径...\MATLAB7.0\toolbox\matlab\general 中，可以查看 mesh.m 文件的程序代码：

```
[mesh.m]
function h = mesh(varargin)
[v6,args] = usev6plotapi(varargin{:});
[cax,args,nargs] = axescheck(args{:});
user_view = 0;
cax = newplot(cax);
hparent = get(cax,'parent');
fc = get(cax,'color');
if strcmpi(fc,'none')
  if isprop(hparent,'Color')
   fc = get(hparent,'Color');
  elseif isprop(hparent,'BackgroundColor')
   fc = get(hparent,'BackgroundColor');
  end
end
[reg, prop]=parseparams(args);
nargs=length(reg);
error(nargchk(1,4,nargs));
if rem(length(prop),2)~=0,
   error('Property value pairs expected.')
end
   if nargs == 1
   x=reg{1};
   if v6
       hh = surface(x,'FaceColor',fc,'EdgeColor','flat', ...
            'FaceLighting','none','EdgeLighting','flat','parent',cax);
   else
       hh = graph3d.surfaceplot(x,'FaceColor',fc,'EdgeColor','flat', ...
            'FaceLighting','none','EdgeLighting','flat','parent',cax);
   end
.................................// 限于篇幅，省略了部分程序代码
```

```
elseif nargs == 4
   [x,y,z,c]=deal(req{1:4});
   if v6
       hh = surface(x,y,z,c,'FaceColor',fc,'EdgeColor','flat', ...
                    'FaceLighting','none','EdgeLighting','flat','parent',cax);
   else
       hh = graph3d.surfaceplot(x,y,z,c,'FaceColor',fc,'EdgeColor','flat', ...
                    'FaceLighting','none','EdgeLighting','flat','parent',cax);
   end
end
if ~isempty(prop),
   set(hh,prop{:})
end
if ~ishold(cax) && ~user_view
   view(cax,3); grid(cax,'on');
end
if nargout == 1
   h = double(hh);
end'
```

从上面的程序代码中，可以看到典型的高层绘图文件的构成：

- 调用命令 newplot，检查并设置图和轴的 NextPlot 属性，然后返回目标轴的句柄；
- 引用 newplot 返回的坐标轴句柄，或者修改坐标轴的属性，或者访问坐标轴的属性；
- 调用图形对象创建命令，创建图形。

坐标轴对象

在图形对象的树形结构中，坐标轴对象（Axes）发挥着重要的作用，其具体的属性有 80 多个。本节将介绍坐标轴对象中一些常见和重要的属性，将会帮助读者更好地设置图形对象的属性。

坐标轴的几何属性

与前面介绍的图形对象类似，轴位框的几何属性主要由 Position、Units 来指定。在 MATLAB 中，轴位框（Axes Position Rectangle）的含义不同于坐标框。对于二维图形，两者指定的是同一个面积；对于二维图形，轴位框指定图形所占用的最大平面面积，而不是三维坐标轴。由于屏幕上没有显示轴位框线，因此这个概念初学者经常会混淆。

在 MATLAB 中，关于坐标轴的“单位”（Units）的默认属性为“normalized”，可以使用 get 函数得到，相应的代码如下：

```
>> get(gca,'units')
ans =
normalized
```

该属性的含义是轴对象使用“归一化”单位。在这种单位下，图形对象窗口的几何属性值总是 [0，0，1，1]，其他对象的单位则使用相对单位。

[illegible]

在 MATLAB 的[illegible]，[illegible]建立[illegible]轴对象，[illegible]轴[illegible]，[illegible]

例 10.12 [illegible]

step 1 [illegible]编辑器[illegible]输入下面的程序代码

```
function [ax,hlines] = plotyyy(x1,y1,x2,y2,x3,y3,ylabels)
%PLOTYYY - Extends plotyy to include a third y-axis

%Syntax:  [ax,hlines] = plotyyy(x1,y1,x2,y2,x3,y3,ylabels)
%
%Inputs: x1,y1 are the xdata and ydata for the first axes' line
%        x2,y2 are the xdata and ydata for the second axes' line
%        x3,y3 are the xdata and ydata for the third axes' line
%        ylabels is a 3x1 cell array containing the ylabel strings

%Outputs: ax -      3x1 double array containing the axes' handles
%         hlines - 3x1 double array containing the lines' handles
[illegible]
   ylabels{1}=' '; ylabels{2}=' '; ylabels{3}=' ';
end
[illegible]
[ax,hlines(1),hlines(2)] = plotyy(x1,y1,x2,y2);
cfig = get(gcf,'color');
pos = [0.1  0.1  0.7  0.8];
offset = pos(3)/5.5;
[illegible]
set(ax,'position',pos);
pos3=[pos(1) pos(2) pos(3)+offset pos(4)];
[illegible]
   'Color','none','XColor','k','YColor','r',...
   [illegible]
hlines(3) = line(x3,y3,'Color','r','Parent',ax(3));
limy3=get(ax(3),'Ylim');
line([limx1(2) limx3(2)],[limy3(1) limy3(1)],...
   [illegible]
[illegible]
set(get(ax(1),'ylabel'),'string',ylabels{1})
[illegible]
[illegible]
```

step 2 单击 M 文件编辑器工具栏的"保存"按钮，将[illegible]，[illegible] MATLAB 的命令窗口中输入下面的代码

```
>> x=[illegible]
[illegible]=x.[illegible];
y2=x;
[illegible]=exp(x);
[illegible]
```

step 3 在命令行窗口，输入程序代码[illegible]，按"Enter"键，得到[illegible]

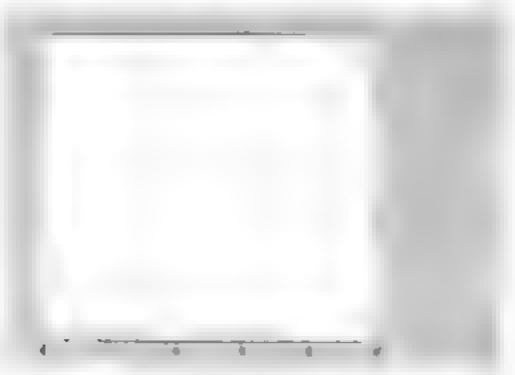


图 10.24 程序所绘制的图形

step 4 单击绘制图形，在MATLAB工作窗口中输入[illegible]的代码

```
[illegible]
>> t1=0:0.01:2*pi;
[illegible]
[illegible]
[illegible]
>> y2=t2.*cos(t2);
>>y3=sqrt(t3).*exp(t3);
>> label=[ 'sin(t)';'t*cos(t)';'sqrt(t)*exp(t)'];
>> [ a,h] =plotyyy(t1,y1,t2,y2,t3,y3,label);
>> set(h,'LineWidth',2)
>> grid
```

step 5 在命令行窗口，输入程序代码[illegible]，按"Enter"键，得到[illegible]

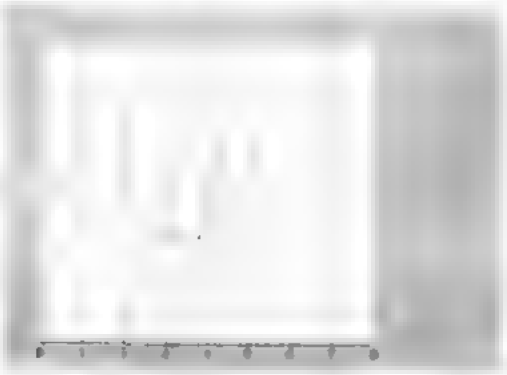


图 10.25 程序重新绘制的图形

在上面的第三段代码中，使用变量pos来定义了一个函数图形的位置，由于第一个函数的坐标采用了LineSet变量，因此第二、三函数坐标轴的右端线。

10.5.2 坐标轴的刻度属性

在MATLAB中，[illegible]，要求修改坐标轴的刻度，则需要调用相应的底层命令

- set(gca,'Xtick',xs,'Ytick',ys)　设置二维坐标刻度
- set(gca,'Xtick',xs,'Ytick',ys,'Ztick',zs)　设置三维坐标刻度。

[illegible]

在MATLAB中，[illegible]坐标轴的刻度属性[illegible]，[illegible]title、mainal[illegible]，[illegible]的文件。

例10.13　在MATLAB中，[illegible]设置坐标轴刻度的属性

step 1　在MATLAB的命令窗口中输入下面的程序代码

```
>>x,[illegible],
>>y2 = 1 ./ ((x2-3).^2 + 1) + 1 ./ ((x2-9).^2 + 4) + 5;
>>plot(x2,y2,'r-','LineWidth',2)
[illegible]
'YtickLabel',{'5.1118';'5.2770';'最大值'});
[illegible]
>>hold on,
>>set(gca,'Ygrid','on')
```

step 2　[illegible]

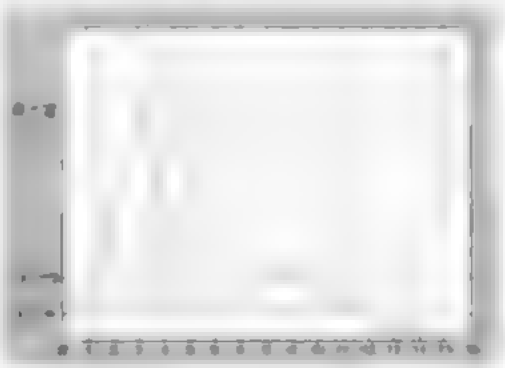


图10.26　设置坐标轴刻度的属性

在MATLAB中，[illegible]

10.5.3 坐标轴的照相机属性

在MATLAB中，可以设置坐标轴的照相机属性。这些属性都是以坐标轴的属性为基础的，坐标轴

的属性可以控制照相机的位置和角度。一般来讲，用户可以使用相应的命令来直接访问坐标轴的照相机属性，表 10.2 列出了其属性的设置和含义。

表 10.2 照相机属性列表

属性	含义
CameraPosition	照相机的位置:[x,y,z]
CameraPositionMode	照相机位置属性的取值模式
CameraTarget	照相机的目标:[x,y,z]
CameraTargetMode	照相机目标属性的取值模式
CameraUpVector	照相机正位向量:[x,y,z]
CameraUpVectorMode	照相机正位向量的取值模式
CameraViewAngle	照相机的视角
CameraViewAngleMode	照相机视角的取值模式
Projection	照相机的投影方式

上面关于照相机的各个属性的物理含义如图 10.27 所示。

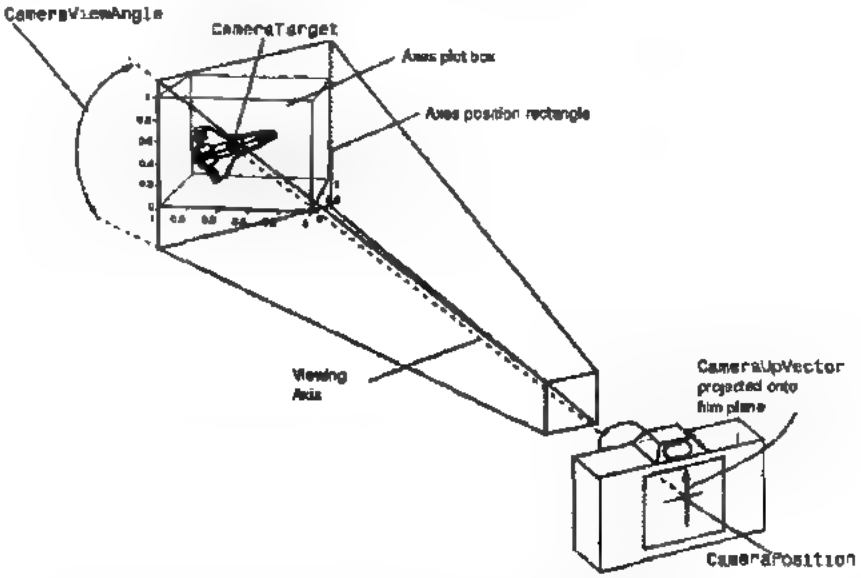


图 10.27 照相机属性的物理意义

在 MATLAB 中，照相机的 CameraTarget 属性在默认情况下的自动取值为坐标框的中心；CameraViewAngle 在自动取值情况下取最小角，使场最张满整个轴位框；同时，照相机的 Projection 属性值为正视投影。

例 10.14 在 MATLAB 中，使用相应的程序代码进行数据可视化，并且选用合适的照相机属性。

step 1 单击 MATLAB 命令窗口工具栏中的 D 按钮，打开 M 文件编辑器。在 M 文件编辑器中输入下面的程序代码.

```
colordef(gcf,'black')

cla
load wind
spd = sqrt(u.*u + v.*v + w.*w);
p = patch(isosurface(x,y,z,spd, 40));
```

[illegible]

Step 2 [illegible]

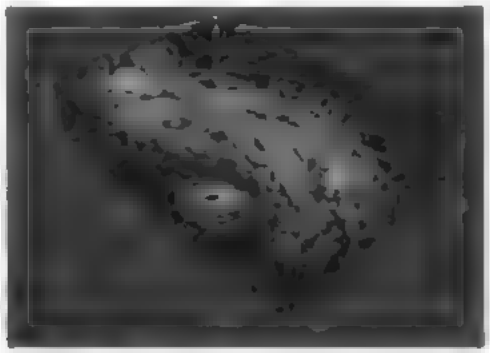


图 10.20 [illegible]

[illegible]

[illegible]

10.6 综合实例

[illegible]

10.6.1 穿越（fly-through）图形

穿越是[illegible]，[illegible]

[illegible]

[illegible]

[illegible]

例 10.15 在 MATLAB 中，绘制 wind 数据文件的穿越图形。

step 1 建立等势表面。在 MATLAB 命令窗口中输入下面的命令

```
>>load wind
>>wind_speed = sqrt(u.^2 + v.^2 + w.^2);
>>hpatch = patch(isosurface(x,y,z,wind_speed,35));
>>isonormals(x,y,z,wind_speed,hpatch)
>>set(hpatch,'FaceColor','red','EdgeColor','none');
```

step 2 查看建模结果。输入[illegible]，按"Enter"键，[illegible]

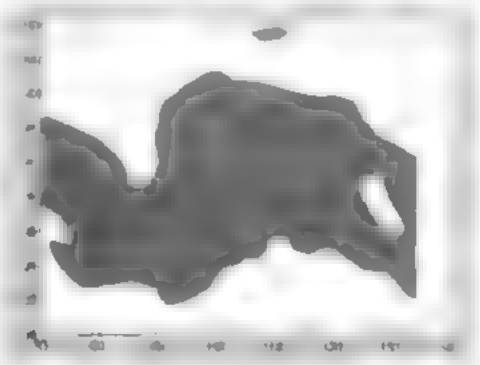


图 10.29 绘制的风速等势表面

step 3 绘制穿越等势表面的锥体。建立[illegible]

```
>>[illegible] = [illegible]isosurface(x,y,[illegible]),[illegible]
>>daspect([1,1,1])
>>hcone = coneplot(x,y,z,u,v,w,vt(:,1),vt(:,2),vt(:,3),2);
>>set(hcone,'[illegible]','[illegible]','[illegible]','[illegible]');
```

step 4 查看[illegible]。输入[illegible]，按"Enter"键，[illegible]

step 5 定义坐标参数。[illegible]视角参数。在 MATLAB 的命令窗口中输入下面的代码

```
>> camproj perspective
```

step 6

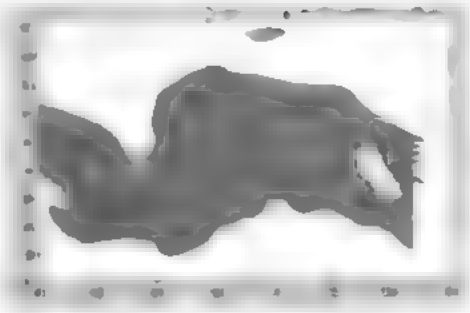


图 10.30 绘制曲面实体

图 10.31 设置视角参数

step 7

```
>> hlight = camlight('headlight');
```

```
>>set(gcf,'Color','k')
```

step 8

图 10.32 设置船形的光源属性

Step 9 选择曲面对象的属性 Renderer 属性，在 MATLAB 的命令窗口中输入下面的代码

```
>> lighting phong
>> set(ans,'renderer','zbuffer')
```

[illegible]

Step 10 定义照明路线。在 MATLAB 的命令窗口中输入下面的代码

```
>> hsline = streamline(x,y,z,u,v,w,80,30,11);
>> xd = get(hsline,'XData');
>> yd = get(hsline,'YData');
>> zd = get(hsline,'ZData');
>> delete(hsline)
```

[illegible]

Step 11 演示穿越效果。在 MATLAB 的命令窗口中输入下面的代码

```
>>for i=1:length(xd)-50
  campos([xd(i),yd(i),zd(i)])
  camtarget([xd(i+5)+min(xd)/100,yd(i),zd(i)])
  camlight(hlight,'headlight')
  drawnow
end
```

Step 12 查看渲染结果。[illegible]键，[illegible]穿越效果。由于是动态效果，[illegible]，使用[illegible]变到动态效果，图形依次如图 10.33 至图 10.35 所示。

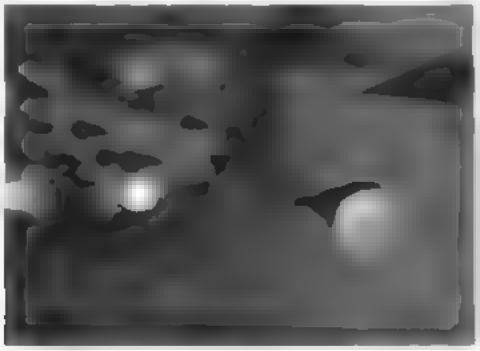


图 10.33 动态图形一

[illegible]

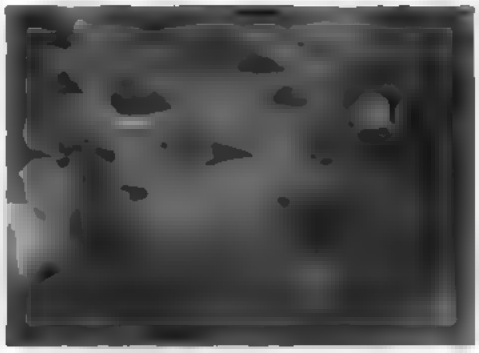


图10.34 动态图形二

图10.35 动态图形三

[illegible]，调用drawnow命令来绘制移动后的图形。

[illegible]窗中绝大部分的像素颜色不变，而只更新部分像素[illegible]

在MATLAB中，drawnow命令[illegible]

10.6.2 动态反射图形

[illegible]

在MATLAB中，[illegible]计算量大，占用内存多，而且播放时间短。

[illegible]实时动画，这种动画比较适合于每次变化较[illegible]EraseMode

在MATLAB中，[illegible]上面的效果便[illegible]，在MATLAB中可以使用系统的EraseMode属性来十分[illegible]的EraseMode属性，其对应的含义如下。

- {normal}：[illegible]
- none：[illegible]
- xor：[illegible]的对象数据点，只擦除与屏幕颜色不一致的数据点。
- background：[illegible]

[illegible]建自己所需的动画效果。

[illegible]

Step 1 绘制初始图形。

Step 2 [illegible]

Step 3 [illegible]

Step 4 重复上面的两个步骤。

[illegible]

例 10.16 在 MATLAB 中，绘制动态光线的反射图形。

[illegible]

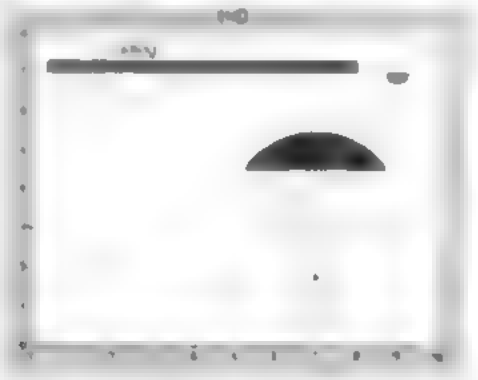


图 10.36 初始图形

[illegible]

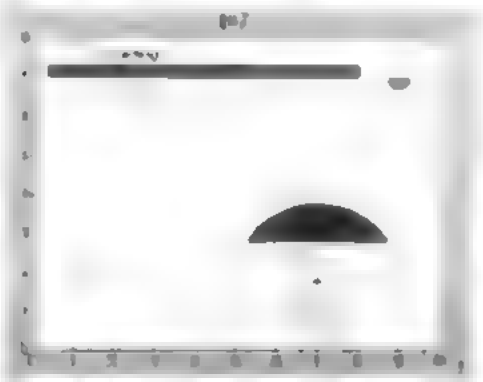


图 10.37 程序的过程图形

[illegible]

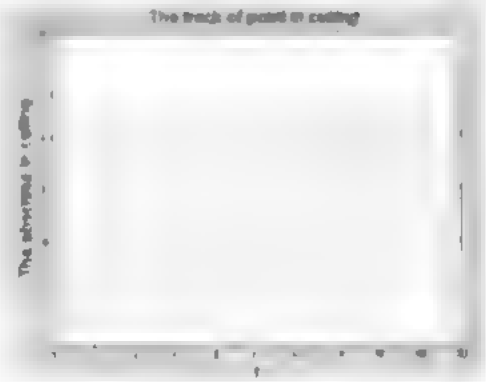


图 10.38　反射点的轨迹

下面将分步骤详细介绍上面的动态效果的创建过程。

step 1 在主MATLAB窗口[illegible]，在M文件编辑器中输入[illegible]的程序代码

```
%---------------设置动态效果的初始环境参数----------------------------%
[illegible]
set(gcf,'DoubleBuffer','on');      %设置绘图的刷新模式
axis([0,10,0,8]);                  %设置坐标轴范围
[illegible]
%---------------创建绘图的顶板(Ceiling)对象----------------------------%
fill([0.4,0.8,0.4],[7,7,7.3,7.3],[0.8,0.1,0.7]);    %创建绘图中的横杆
text(2,7.6,'(\lt\b(ceiling)) ','fontsize',12);      %添加横杆的文字说明
```

step 2 完成[illegible]，按快捷键"F5"，查看该部分代码的运行结果。如图 10.39 所示。

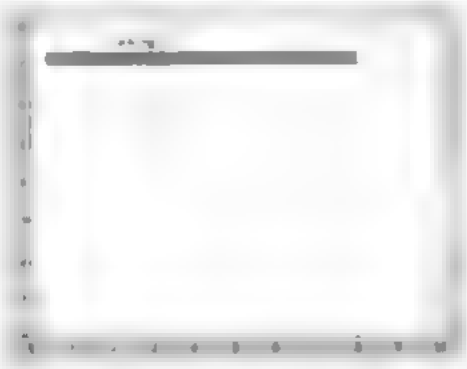


图 10.39　部分程序结果

[illegible]

[illegible]

Step 3 [illegible]

```
[illegible]
set(Fc,'EdgeColor','r');                    %设置光源的边缘颜色
```

Step 4 [illegible]

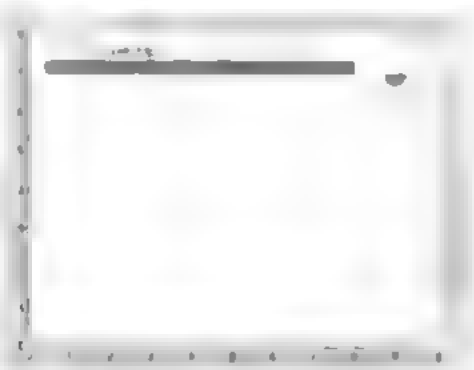


图 10.40 添加半圆光源

说明 [illegible]

Step 5 [illegible]

```
%----------------创建反射半镜对象----------------------------------------
Ar=pi/6;
[illegible]
Fg=fill(real(Zx),imag(Zx),'b');
set(Fg,'EdgeColor','b');
```

Step 6 [illegible]

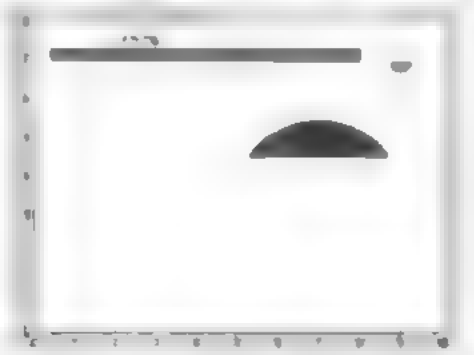


图 10.41 添加反射半镜的对象

[illegible]

Step 7 [illegible]

[illegible]

step 8 [illegible]

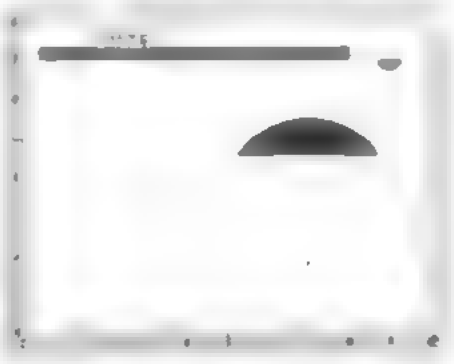


图 10.42 添加球心对象

[illegible]的新点线和横坐标的交点坐标。

step 9 [illegible]

```
%--------------定义发射光线的数学方程----------------------------------%
Dtan=abs(angle(p1-Zx)-angle(Zx-p3));                    %计算相角差
[ a,Kk] =min(Dtan);                                     %计算相角差的最小值
[illegible]                                             %选择放射半圆中的数据点
Hp1=plot([ p1,p2] ,'r');
Hp2=plot([ p3,p2] ,'k:');
```

step 10 [illegible]

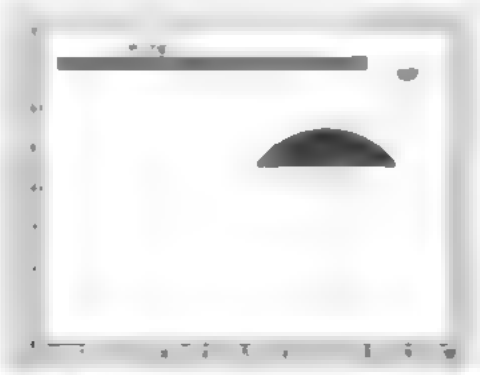


图 10.43 添加发射光线对象

[illegible]数的相角。最后，计算两个相角的差值。

上面步骤计算得到的结果值是一个数组，然后使用min函数来解决相角差的最小值和数组的序号kk。接着，选择反射半圆数据数组[zx,zy]中对应序号kk中的数值pz，在上面程序的最后，绘制发射光线和发射光线的弧长线。

step 11 返回M文件编辑器，在上面的代码后面输入如下代码

```
%-------------定义反射半圆和反射光线的数学方程--------------------------------
At=2*angle(p2-2z)-angle(p2-pz);
L=(1-abs(pz))*cos(At-pi/2);
pf=pz+L*exp(i*At);
Hp3=plot([p2,pf],'r');
```

step 12 查看绘图结果，输入程序代码后，按"Enter"键，绘制的结果如图10.44所示。

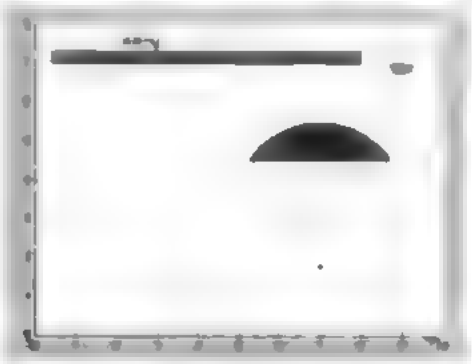


图 10.44 绘制反射光线对象

在上面的程序代码中，首先计算相角At和复数长度L，然后以变量pz为基础变量，计算反射光线在方板中的数据的坐标数值，最后使用plot绘制反射光线。

step 13 返回M文件编辑器，在上面的代码后面输入如下代码

```
%---------------定义推箱的数学方程以及坐标数值---------------------------------
N=16;
xt=ones(1,N);
yt=linspace(0,4.5,N);
Kt=mod(1:N-4,2)-0.5;
xt=xt+[0,0,Kt,0,0];
hpt=plot(xt,yt);
```

step 14 查看图形结果，输入程序代码后，按"Enter"键，绘制的结果如图10.45所示。

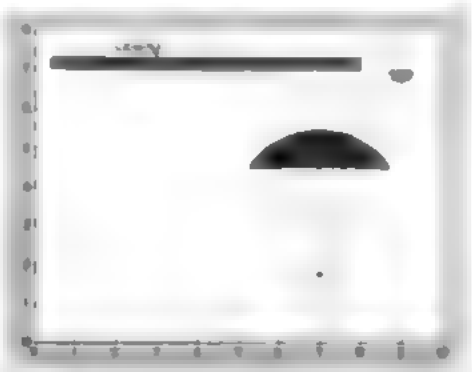


图 10.45 添加弹簧对象

在上面的代码中，[illegible]数，[illegible]数，[illegible]数，[illegible]mod函数的用法请查阅数值计算的章节。

step 15 返回M文件编辑器，在上面的代码后面输入程序代码

```
%--------------- 定义图形的初始标题和时间标量 ----------------------------%
hw=title('t=0','fontsize',16);
text(0.4,2,'t\in [0,20]','fontsize',14,'color',[0.2,0.8,0.6]);
```

step 16 [illegible]

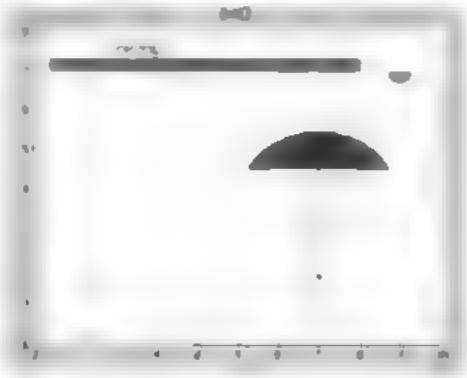


图 10.46 添加图形的标题和说明

[illegible]

[illegible]

step 17 [illegible]

```
%--------------- 定义图形动画的初始参量 ----------------------------%
t=0;              % 控制时间
pg=real(pf);      % 记录反射点的轨迹
dt=0.04;          % 时间的单位步长量
pause(0.1);       %  做一停顿
%--------------- 定义图形动画的循环程序代码 ----------------------------%
while t<20;
    t=t+dt;
    [illegible]
    Zx=pd-2*cos(Ar)*i+2*exp(i*linspace(Ar,pi-Ar,60));
    [illegible]
    Zr=pd-2*cos(Ar)*i;
    set(hc,'XData',real(Zr),'YData',imag(Zr));
    [illegible]angle(Zx-pd);
    [a,Kk]=min(Dtan);p2=Zx(Kk);
```

```
    set(Hp1,'XData',real([p1,p2]),'YData',imag([p1,p2]));
    [illegible]
    Af=2*angle(p2-&r)-angle(p1-p2);
    [illegible]
    t=[illegible]exp([illegible]
    pg=[pg,real(pf)];
    [illegible]
    [illegible]
    [illegible]
    [illegible]
    set(hw,'String',['t=',num2str(t)]);
    [illegible]
end
```

Step 18 [illegible]

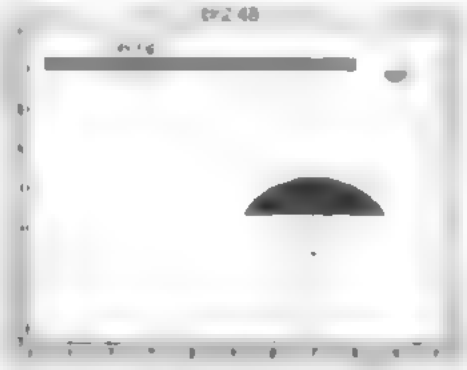


图 10.47 t = 2.48时的图形界面

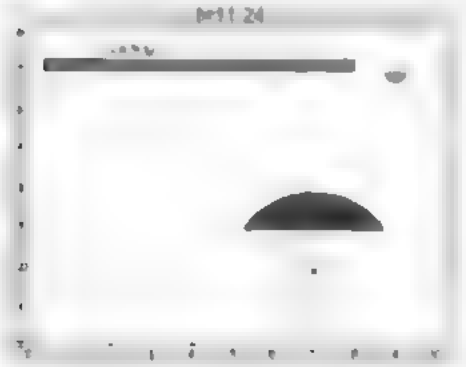


图 10.48 t = 11.24时的图形界面

从上面的程序结果可以看出，在MATLAB环境下，图形句柄将产生[illegible]秒，最后的结果在前面的叙述中已经给出，这里就不重复了。

[illegible]

Step 19 返回M文件编辑器，在上面的代码后面输入程序代码

```
%-------------------创建一个新的图形窗口--------------------------------%
figure;
[illegible]
title('The track of point in ceiling','fontsize',14);
xlabel('t\ttt','fontsize',14);
ylabel('The abscissa in ceiling','fontsize',14)
```

Step 20 [illegible]

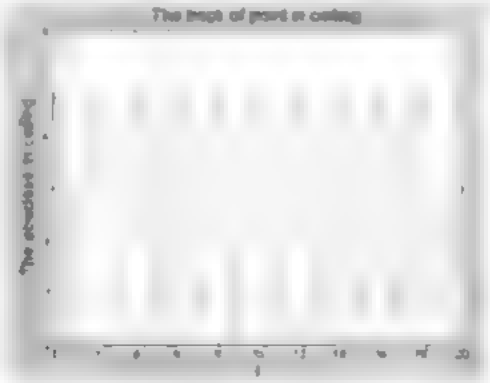


图 10.49　反射点轨迹图形

Step 21 [illegible]

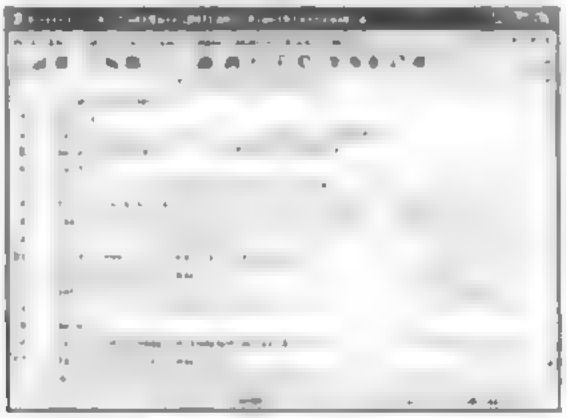


图 10.50　保存代码程序

[illegible]

10.7 小结

[illegible]

◆　◆　◆

第11章　图形用户界面（GUI）制作

本章概述

- 使用GUIDE创建GUI
- 使用M文件创建GUI
- 创建自定义菜单
- 创建现场菜单
- 添加GUI的图形控件

[illegible]

11.1　图形用户界面概述

[illegible]

在MATLAB中，[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

文件包含了GUI初始化代码和组建界面布局的控制代码。

[illegible]

对应文件。

- **FIG文件**：[illegible]
- **M文件**：[illegible]文件不包含用户界面设计的代码，对应代码出FIG文件保存。

[illegible]

11.2 使用M文件创建GUI对象

[illegible]节中，将使用两个简单的实例来说明各种方法的特点。

[illegible]编写而不适合使用GUIDE来创建。

[illegible]

11.2.1 编写程序代码

例11.1 [illegible]

控制齿轮运动的方向，使用方向按键来控制查看该二维对象的角度。

step 1 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的"File" ⇨ "New" ⇨ "M-file"命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function gear3d(varargin)
%   GEAR3D   GUI example of 3D gear.
%   GEAR3D will pop up a GUI example of 3D gear. The gear rolls on a
geared
%   ground based on the mouse location. It uses the x-location of the
mouse
%   pointer for the gear location. Use the arrow keys to change the
view.
%   Press SPACEBAR to reset the view.
%
%
%   调用命令示例
%     GEAR3D('teeth', 30)   - 默认值是 50
%     GEAR3D('spokes', 4)   - 默认值是 8
%     GEAR3D('ratio', 2)    - 默认值是 3
%   该命令只接受正值的参数

% ------------------------参数的默认数值-----------------------------%
% Number of teeth
numteeth  = 50;
% Number of spokes
numspokes = 8;
% Gear ratio
rr        = 3;
% ------------------------处理函数的参数属性-----------------------------%
if mod(nargin, 2) == 1
  error('Optional arguments must come in pairs.');
end
if nargin
  opt = varargin(1:2:end);
  val = varargin(2:2:end);
  validOpts = {'teeth', 'spokes', 'ratio'};
  for iArg = 1:length(opt)
    id = strmatch(lower(opt{iArg}), validOpts);
    if isempty(id)
      error('Invalid option. Valid options: ''teeth'', ''spokes'',
''ratio''');
    else
      switch strmatch(lower(opt{iArg}), validOpts)

        case 1
          if isnumeric(val{iArg}) & length(val{iArg}) == 1
            numteeth = round(abs(val{iArg}));
          end
                  case 2
          if isnumeric(val{iArg}) & length(val{iArg}) == 1
            numspokes = round(abs(val{iArg}));
          end
                  case 3
          if isnumeric(val{iArg}) & length(val{iArg}) == 1
```

```
            rr = round(abs(val(iArg)));
          end
            end
    end
  end
end
%-----------------------------------计算参数的数值-----------------------------%
% 半径数值
r1 = 1;
r2 = 3;
r3 = 10;
r4 = 11;
r5 = 13;
r6 = 12;
% 中心齿轮的半径数组合 i
r = (r5 + r6) / 2;
% Height of gear teeth
h = r5 - r6;

l = r * rr;
l1 = l  - h/2;
l2 = l  + h/2;
l3 = l2 + 2;
%-----------------------------定义齿轮对象的数据-----------------------------%
[x0,y0,z0]        = cylinder([r6], numteeth*4);  % 齿轮的凹槽
[x1,y1,z1]        = cylinder([r1, r1, r2, r2, r1], numteeth*4);  % 轮辐
[x2,y2,z2]         = cylinder([r3, r3, r4, r4, r5, r5, r4, r4, r3],
numteeth*4);  % 轮齿
z1([1, 4, 5], :) =  2;
z1(2:3, :)       = -2;
z2([1, 8, 9], :) =  2;
z2(2:3, :)       = -2;
z2(4:5, :)       = -1;
z2(6:7, :)       =  1;
x2(5:6,1:4:end)  = x0(1:2,1:4:end);
x2(5:6,2:4:end)  = x0(1:2,2:4:end);
y2(5:6,1:4:end)  = y0(1:2,1:4:end);
y2(5:6,2:4:end)  = y0(1:2,2:4:end);
%-----------------------------定义轮辐对象的数据-----------------------------%
interval  = round(length(x1) / numspokes);
for id = 1:4
  x1(3:4, id:interval:end) = x2(1:2, id:interval:end);
  y1(3:4, id:interval:end) = y2(1:2, id:interval:end);
end
%-----------------------------定义底面对象的数据-----------------------------%
[x3, y3, z3]         = cylinder([l2, l2, l3, l3, l2], numteeth * rr * 4);
[x4, y4, z4]          = cylinder(l1, numteeth * rr * 4);
z3([1 4 5], :)        =  4;
z3([2 3], :)          = -4;
x3([1 2 5], 1:4:end) = x4([1 1 1], 1:4:end);
x3([1 2 5], 2:4:end) = x4([1 1 1], 2:4:end);
y3([1 2 5], 1:4:end) = y4([1 1 1], 1:4:end);
y3([1 2 5], 2:4:end) = y4([1 1 1], 2:4:end);
%-----------------------------将底面设置为半圆-----------------------------%
len             = round(length(x3) / 2);
x3(:, len:end) = [];
```

```
y3(:, len:end) = [];
z3(:, len:end) = [];
```

[illegible]

[illegible]

step 2 [illegible]

```
%----------------------------创建图形对象----------------------------%
fH = findobj('Type', 'figure', 'Tag', 'solidmodelGUI');
[illegible]
  [illegible];
[illegible]
%创建图形对象的位置
fH = figure(...
  'Name'        , [illegible], ...
  'NumberTitle', 'off',
  'Units'       , 'normalized', ...
  'Position'    , [illegible], ...
  'Tag'         , [illegible]);
%设置图形对象的坐标轴属性
axes(...
  'Units'    , 'normalized', ...
  'Position', [0, .9, 1, .1], ...
  [illegible]
%添加文字提示信息
text([illegible]
  ['Move the [illegible] left/right within the [illegible] window [illegible] the
gear.',...
    'Use ARROW keys to change the view. SPACE to reset view.'), ...
  'HorizontalAlignment'  , [illegible],
  'VerticalAlignment'    , 'middle',
  [illegible]
[illegible] = axes([illegible], ...
  'Position', [0, 0, 1, .9]);
[illegible]

[illegible]
gearH(1) = surface(x1,y1,z1, [illegible]
  'EdgeColor', [.3, .3, .3],
  'EdgeAlpha', [illegible],
  'FaceColor', [.5, [illegible]
[illegible]-绘制轮齿----------------------------%
gearH(2) = surface(x2,y2,z2,
  'EdgeColor', [.3, .3, .3], ...
  'EdgeAlpha', .1,
```

```
  'FaceColor', [ .5, [illegible], 1]);
%-[illegible]---绘制曲面---------------------------------------%
surface(x3,y3,z3,
  [illegible]
  [illegible]
  [illegible]
  [illegible]
  [illegible]
%-----------------------------设置图形的视角-----[illegible]
set(axH, 'View', [0, -50]);
[illegible]
set(axH, 'CameraViewAngleMode', 'manual');
%-----------------------------添加光源---------------------------------%
light('Position', [50 , 0, -20]);
light('Position', [-50, 0, -20]);
```

step 3 [illegible]

图 11.1 GUI 的初始图形效果

[illegible]

step 4 返回 M 文件编辑器，输入下面的代码

```
%-----------------------------计算系统大小并传递参数-----------------------%
sz1 = size(x1);
sz2 = size(x2);
[illegible]
%-[illegible]--设置法线的初始位置-[illegible]
[illegible]

%----------------myKeyPressFcn 函数，单击方向键改变视角----------------%
[illegible]
```

```
k = get(obj, 'CurrentKey');
aZeL = get(axH, 'View');

switch k, [illegible]
  case 'uparrow'
    aZeL(2) = min([aZeL(2) + 10,  90]);
  case 'downarrow'
    % -90 doesn't work well when rotated left or right
    aZeL(2) = max([aZeL(2) - 10, -89.9999]);
  case '[illegible]'
    aZeL(1) = max([aZeL(1) - 10, -90]);
  case 'rightarrow'
    aZeL(1) = min([aZeL(1) + 10,  90]);
  case 'space'
    aZeL    = [0, -50];
end
set(axH, 'View', aZeL);
set(obj, 'Name', sprintf('3D Gear: View = [%3.0f, %3.0f]', aZeL));
%-------------------------myMotionFcn函数 生成动画效果-------------------------%
function myMotionFcn(obj, edata, gearH, x1, x2, y1, y2, r, d, sz1, sz2)
pt = get(obj, 'CurrentPoint');
d = 1 - r;
x0 = -d + pt(1) * d * 2;
y0 = sqrt(abs(d^2 - x0^2));
th1 = atan2(x0, y0) - pi/2;
th = (th1 - (th1 * (1 / r)));
%-------------------------[illegible]-------------------------%

%-------------------------设置旋转角度矩阵-------------------------%
cosa = cos(-th);
sina = sin(-th);
rot  = [cosa, -sina; sina, cosa]';
newxy1   = [x1(:), y1(:)];
newxy2   = [x2(:), y2(:)];
newxy1   = newxy1 * rot;
newxy2   = newxy2 * rot;
newx1    = x0 + reshape(newxy1(:, 1), sz1);
newy1    = y0 + reshape(newxy1(:, 2), sz1);
newx2    = x0 + reshape(newxy2(:, 1), sz2);
newy2    = y0 + reshape(newxy2(:, 2), sz2);
%-------------------------设置对象的新位置-------------------------%
set(gearH, {'XData', 'YData'}, {newx1, newy1; newx2, newy2});
```

说明 在上面的程序代码中，[illegible]，在各个程序代码之间都加入了对应的注释文字，[illegible]，读者自行分析。

11.2.2 运行程序代码

接续上面小节的步骤。

Step 1 在命令窗口中输入[illegible]，运行[illegible]，选择M文件编辑器[illegible]，[illegible]

图 11.2　使用鼠标控制齿轮的运动

[illegible]

Step 2 [illegible]角，如图 11.3 所示。

[illegible]

图 11.3　使用方向键控制视角

Step 3 [illegible]'ratio', 4)" 命令，然后按 "Enter" 键，得到的图形如图 11.4 所示。

[illegible]参数值，然后根据对应的参数绘制图形。

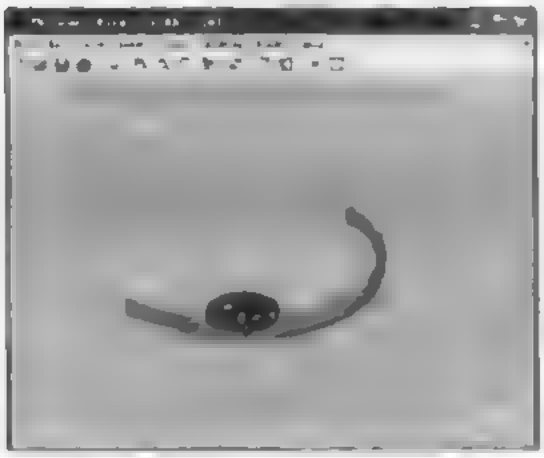


图 11.4 使用自定义参数绘图

[illegible]

11.3 使用 GUIDE 创建 GUI 对象

[illegible]

本节将使用一个简单实例来介绍如何使用 GUIDE 创建 GUI 对象。

11.3.1 启动 GUIDE

在本小节中，将使用 GUIDE 来定制 GUI 对象的布局，使用 M 文件来编写响应事件的事件处理程序，以完成整个 GUI 的创建工作。该 GUI 的最终结果如图 11.5 所示。

图 11.5 完成的 GUI 对象

[illegible]，[illegible]系统中输入一维函数的[illegible]，绘制函数[illegible]，同时可以选择输出计算的结果。

[illegible]

下面分步骤来介绍如何创建该 GUI 对象。

例 11.2 在 MATLAB 中，创建上面的 GUI 对象。

Step 1 [illegible]，在 MATLAB 工作界面，[illegible]，[illegible] MATLAB 中的 GUIDE，如图 11.6 所示。

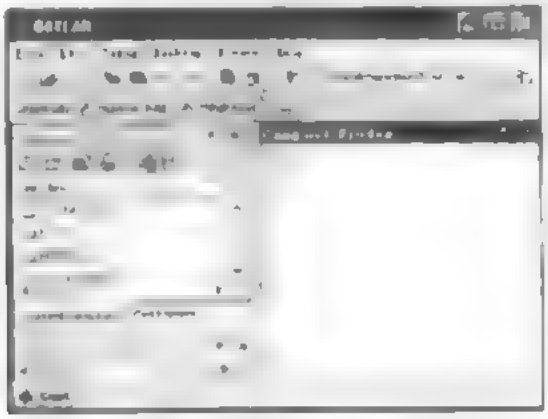


图 11.6 启动 GUIDE

Step 2 [illegible]，如图 11.7 所示。

[illegible]

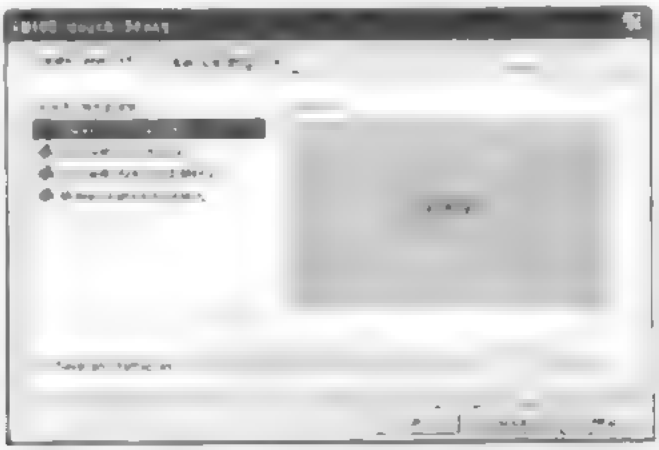


图 11.7 GUI 模板设置界面

[illegible]

Step 3 [illegible] 模板，在 "GUIDE Blank GUI" 对话框中选择 [illegible] 选项，单击 "OK" 按钮，MATLAB 会显示 GUIDE 初始化对话框，如图 11.8 所示。

图 11.8 初始化对话框

[illegible]

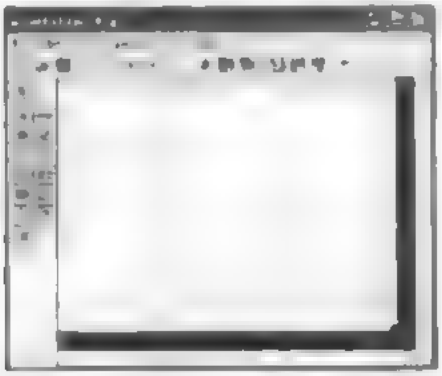


图 11.9 空白模板的编辑界面

Step 4 [illegible] "Preferences" 对话框，[illegible] 选项，选择 "Show names in component palette" 选项，显示各个控件的名称，如图 [illegible]。

Step 5 查看模板的属性，单击 "Preferences" 对话框中 "OK" 按钮，[illegible] 模板，如图 11.11 所示。

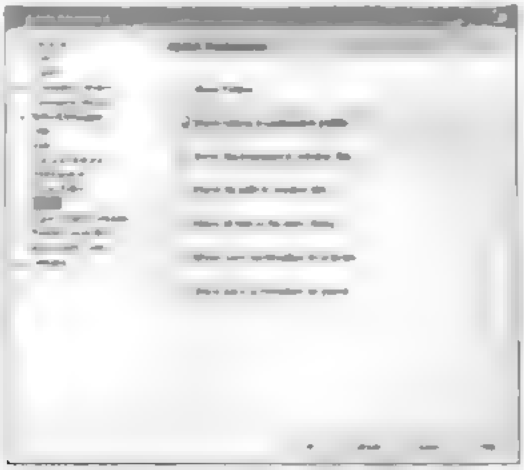


图 11.10 设置模板的显示属性

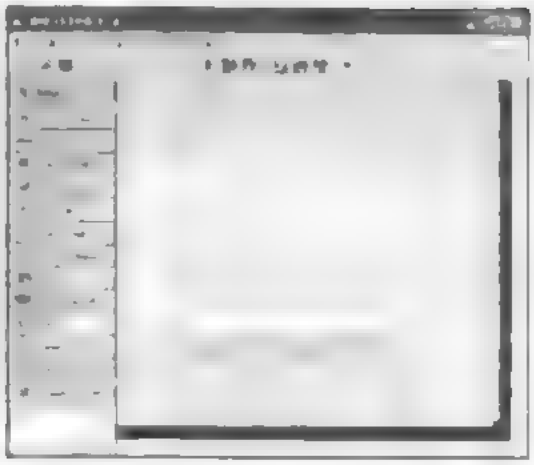


图 11.11　修改后的空白模板

[illegible]

11.3.2 添加控件组件

[illegible]按照上面小节的步骤：

Step 1 [illegible]中合适的位置上。得到的结果如图 11.12 所示。

[illegible]坐标轴控件设置外观和行为的参数。

[illegible]

Step 2 [illegible]提示。

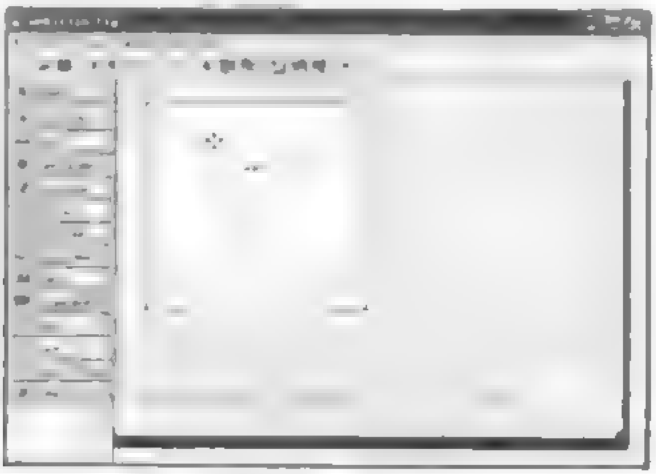


图 11.12　添加"坐标轴"控件

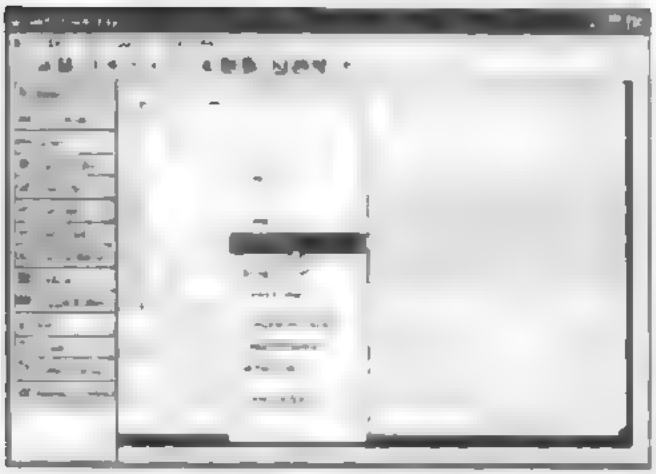


图 11.13 复制"坐标轴"控件

step 3 在[illegible]的"坐标轴"控件[illegible]"复制"坐标轴"控件[illegible]，[illegible]移动复制后控件的位置，得到的结果如图 11.14 所示。

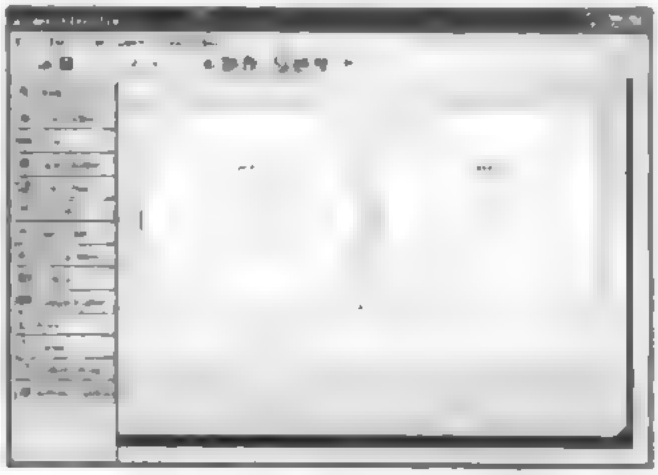


图 11.14 复制后的控件

step 4 添加"组合框（Panel）"控件[illegible]，需要[illegible]参数，为了便于管理，需要将这些参数放置在组合框控件中。[illegible]"Panel"控件，然后将其[illegible]面板中，如图 11.15 所示。

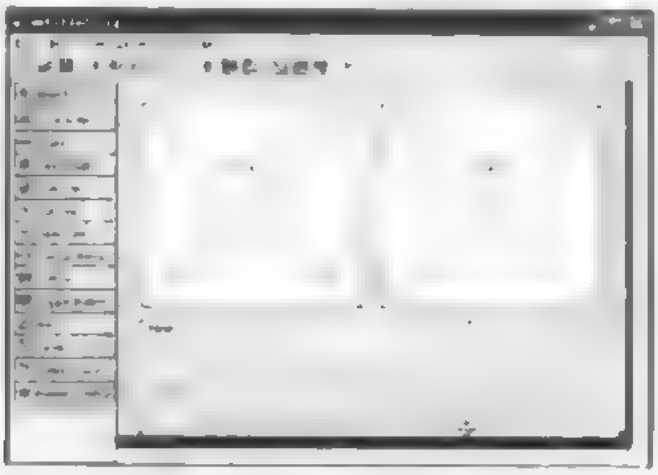


图 11.15 添加组合框控件

[illegible]

[illegible]

Step 5 [illegible]添加到上面步骤中添加的"Panel"对象中，如图 11.16 所示。

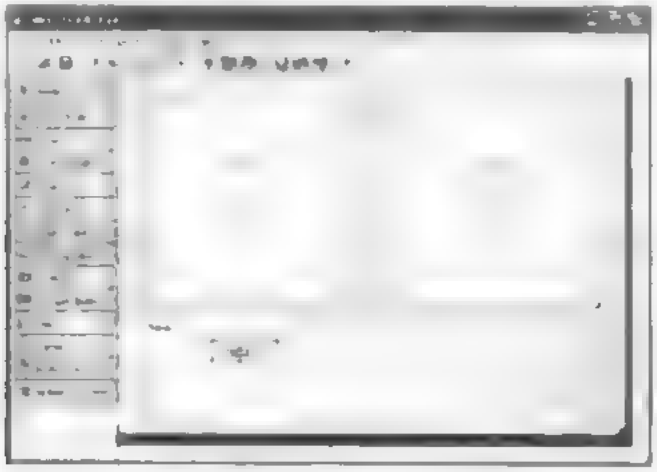


图 11.16　添加编辑框控件

[illegible]

Step 6 [illegible]

图 11.17　添加静态文本控件对象

[illegible]

step 7 [illegible]

图 11.18 复制编辑组和静态文本控件

step 8 [illegible]面板中，如图 11.19 所示。

[illegible]现复杂行为并调用相应的回调子函数。

step 9 [illegible]件，因此在本步骤中需要复制该按钮控件，如图 11.20 所示。

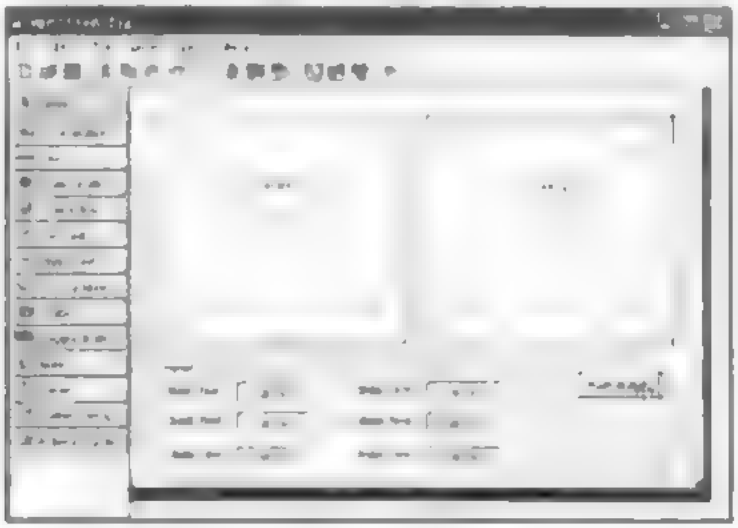


图 11.19 添加按钮控件

图 11.20　复制按钮控件

11.3.3　设置控件组件的属性

接续上面小节的步骤：

step 1　设置各个界面的属性，选中第一个组件，单击"Property Inspector"按钮，打开"Property Inspector"对话框，然后选择"Name"选项，在其中输入标题"Vehicle Simulator"。如图 11.21 所示。

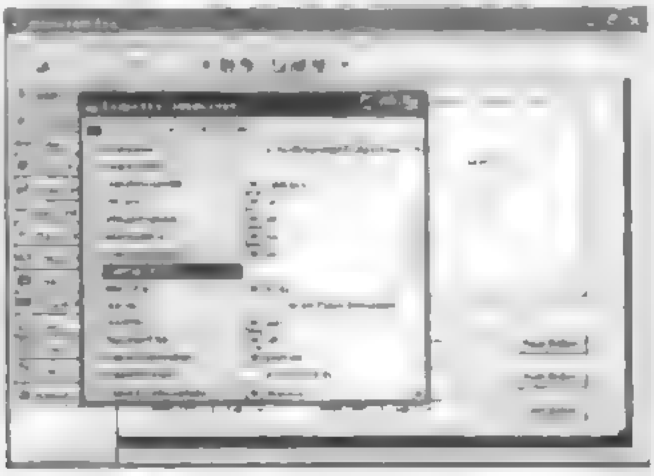


图 11.21　设置图形界面的标题

step 2　设置坐标轴控件的属性，选中"Axes"控件，单击"Property Inspector"按钮，打开"Property Inspector"对话框，在其中设置坐标轴的属性，如图 11.22 所示。

在上面的"Property Inspector"对话框中，选择"Height/Width"选项，将其设置为默认，然后选择"Visible"选项，将其设置为"off"，选择"Title"选项，将其设置为空白，也就是删除标题，其他属性保持默认属性。

step 3　设置"静态文本"控件的属性，选择第一个"静态文本"控件，单击"Property Inspector"按钮，打开"Property Inspector"对话框，选择"Name"选项，在其中输入"X component"，得到的结果如图 11.23 所示。

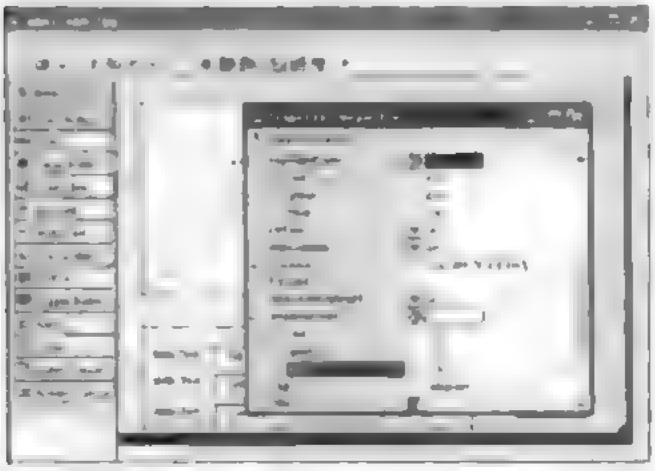


图 11.22 设置组合框控件的属性

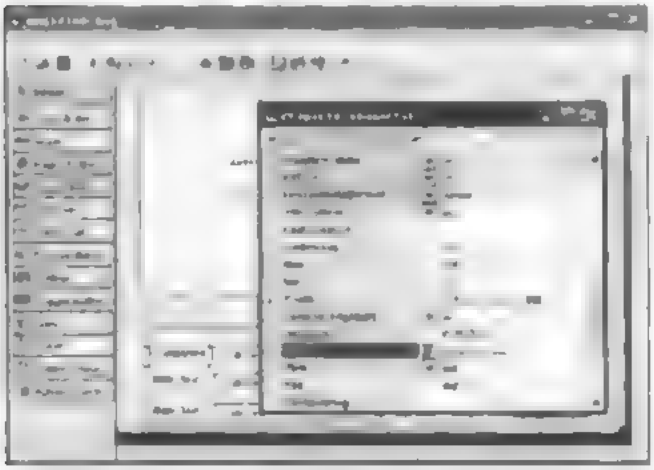


图 11.23 设置"静态文本"的标题

step 4 依次修改其他"静态文本"控件的文字。可以使用和上面步骤类似的方法来修改其他"静态文本"控件的文字，修改后的结果如图 11.24 所示。

step 5 设置左侧"编辑框"控件的属性。在本实例中，左侧的"编辑框"控件提供给用户输入数据表达式，因此该编辑框的默认数值是"空格"。选择第一个"静态文本"控件，然后单击"Property Inspector"按钮，打开"Property Inspector"对话框，选择"String"选项，将其设置为"空"，得到的结果如图 11.25 所示。

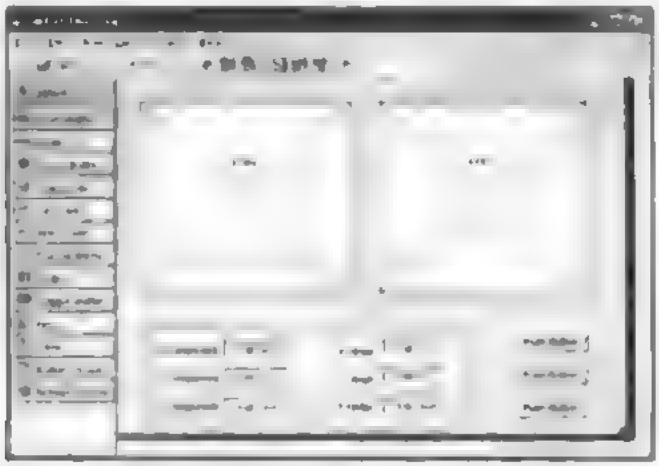


图 11.24 修改其他"静态文本"控件的文字

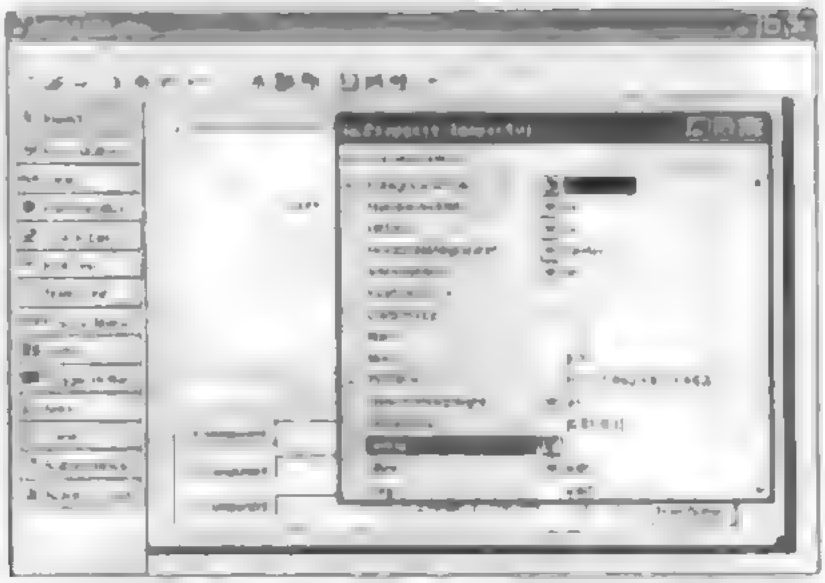


图 11.25　设置左侧编辑框控件的属性

说明

[illegible]

Step 6　[illegible] "[−1,1]"，得到的结果如图 11.26 所示。

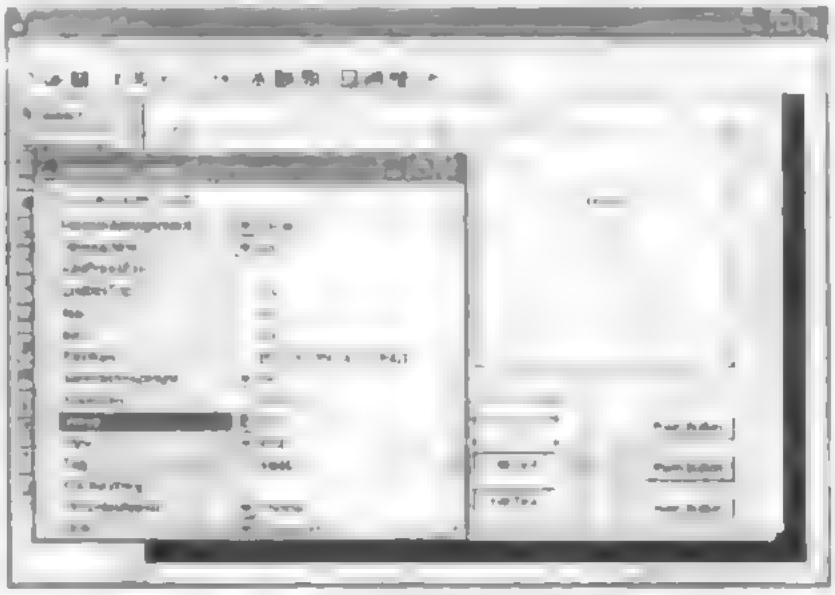


图 11.26　设置右侧编辑框控件的属性

Step 7　[illegible] 度，以及对齐方式，得到的最后结果如图 11.27 所示。

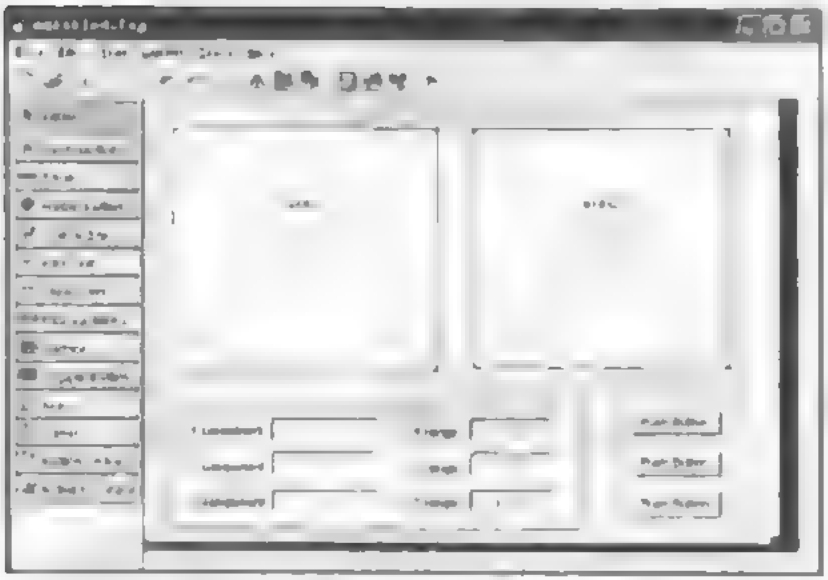


图 11.27 设置编辑框属性后的结果

step 8 设置"按钮"控件的名称。选择"按钮"控件，单击"Properties Inspector"按钮，打开"Properties Inspector"对话框，选择"Name"选项，设置为"Draw"，如图 11.28 所示。

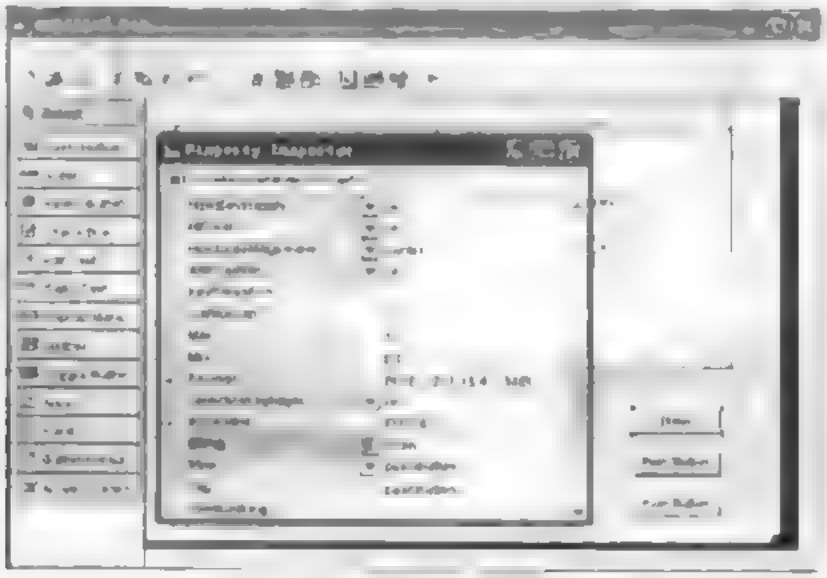


图 11.28 设置"按钮"控件的名称

说明 修改设置为相应的"按钮"控件的名称，例如为"Draw"和"Clear"，以便于在后面为其编写相应的事件处理函数。

step 9 设置"按钮"控件的属性。选择"按钮"控件，单击"Properties Inspector"按钮，打开"Properties Inspector"对话框，选择"Label"选项，选择为"Draw"，如图 11.29 所示。

step 10 全局属性设置的结束。回到主菜单"File"，选择"Save"菜单项，打开"Save"对话框，选择合适的目录，单击"Save"按钮，保存属性设置的结果，如图 11.30 所示。

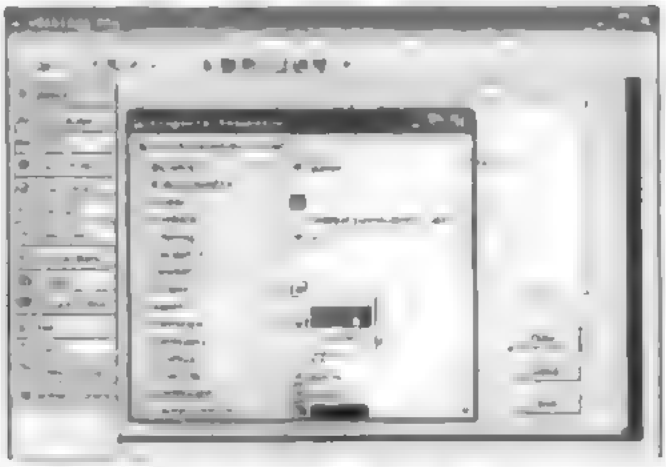


图 11.29 设置“按钮”控件的属性

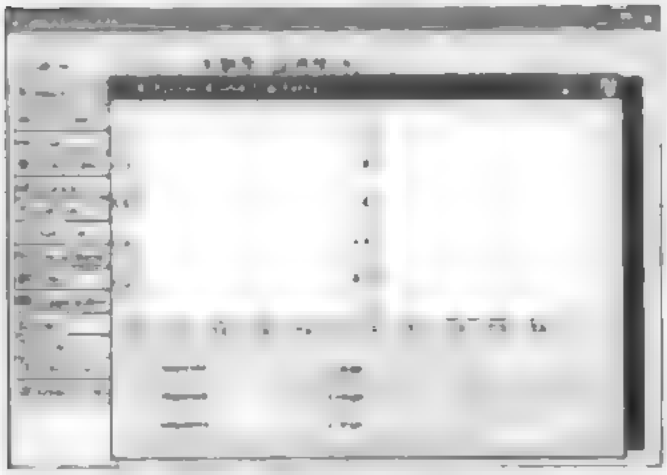


图 11.30 查看属性设置的结果

[illegible]

11.3.4 编写相应的程序代码

接续上面小节的步骤。

step 1 设置各控件的“Tag”属性。选择“Draw”按钮，单击“Property Inspector”按钮，打开“Property Inspector”对话框，选择“Tag”选项，设置为“Draw”，如图 11.31 所示。

重复上面步骤，[illegible]

- 编辑框 [illegible] Tag 属性为“t_comp”
- 编辑框 [illegible] Tag 属性为“y_comp”
- 编辑框 [illegible] Tag 属性为“t_range”
- 编辑框 [illegible] Tag 属性为“y_range”
- 编辑框 [illegible] Tag 属性为“y_range”
- 编辑框 [illegible] Tag 属性为“t_range”
- 按钮 [illegible] Tag 属性为“output”
- 按钮 [illegible] Tag 属性为“close”

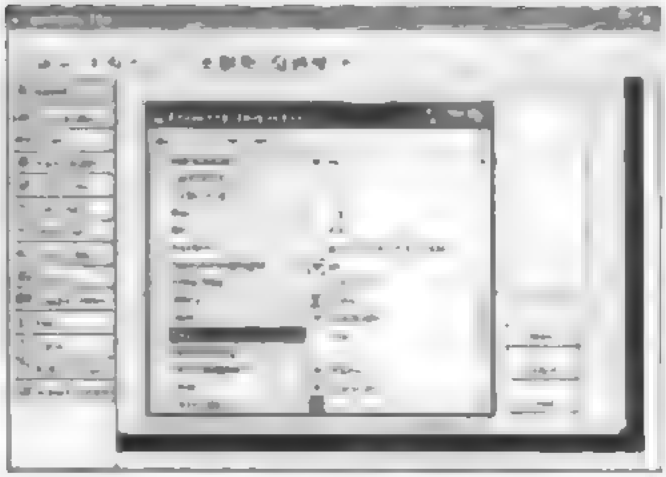


图 11.31 设置控件的"Tag"属性

[illegible]

Step 2 [illegible] M 文件编辑器，[illegible]"Tag"—"M-file editor"[illegible]，[illegible]单击菜单栏的字"M-file editor"按钮，[illegible] M 文件编辑器，如图 11.32 [illegible]

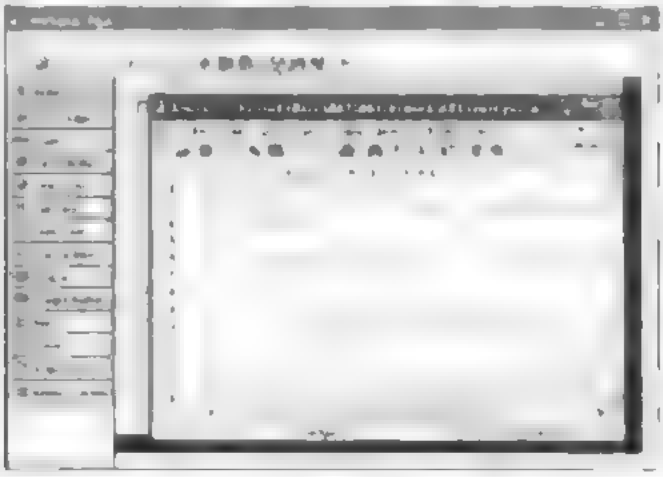


图 11.32 打开 M 文件编辑器

Step 3 在新 M 文件[illegible]，[illegible]自动产生的[illegible]，[illegible]编辑 M 文件[illegible]其部分代码程序如下

```
function varargout = vectgui(varargin)
%   VECTGUI M-file for vectgui.fig
%      VECTGUI, by itself, creates a new VECTGUI or raises the existing
%      singleton*.
%      H = VECTGUI returns the handle to a new VECTGUI or the handle to
%      the existing singleton*.
%      VECTGUI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in VECTGUI.M with the given input arguments.
%      VECTGUI('Property','Value',...) creates a new VECTGUI or raises the
```

```
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before vectgui_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to vectgui_OpeningFcn via varargin.
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
% See also: GUIDE, GUIDATA, GUIHANDLES
% Copyright 2002-2003 The MathWorks, Inc.
% Edit the above text to modify the response to help vectgui
% Last Modified by GUIDE v2.5 17-Jul-2006 11:02:59
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @vectgui_OpeningFcn, ...
                   'gui_OutputFcn',  @vectgui_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before vectgui is made visible.
function vectgui_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to vectgui (see VARARGIN)
% Choose default command line output for vectgui
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
% UIWAIT makes vectgui wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = vectgui_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;

function Y_Comp_Callback(hObject, eventdata, handles)
% hObject    handle to Y_Comp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of Y_Comp as text
[illegible]
as a double

% --- Executes during object creation, after setting all properties.
function Y_Comp_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Y_Comp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc
    set(hObject,'BackgroundColor','white');
else
[illegible]
end
[illegible]//略去部分代码
% --- Executes on button press in Close.
function Close_Callback(hObject, eventdata, handles)
% hObject    handle to Close (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

[illegible]部分的结构。

- **函数名称**：[illegible]
- **函数的注释文字**：[illegible]
- **GUI 对象的初始化代码**：[illegible]程序代码，这部分代码用户不能编辑（DO NOT EDIT）。
- **函数 vectgui_OpeningFcn 的程序代码**：[illegible]
- **函数 vectgui_OutputFcn 的程序代码**：[illegible]
- **控件的创建函数**：[illegible]添加的所有控件，MATLAB 都会设置对应的创建函数。
- **控件的回调函数**：[illegible]部分。

[illegible]

step 6 [illegible] 命，如图 11.33 所示。

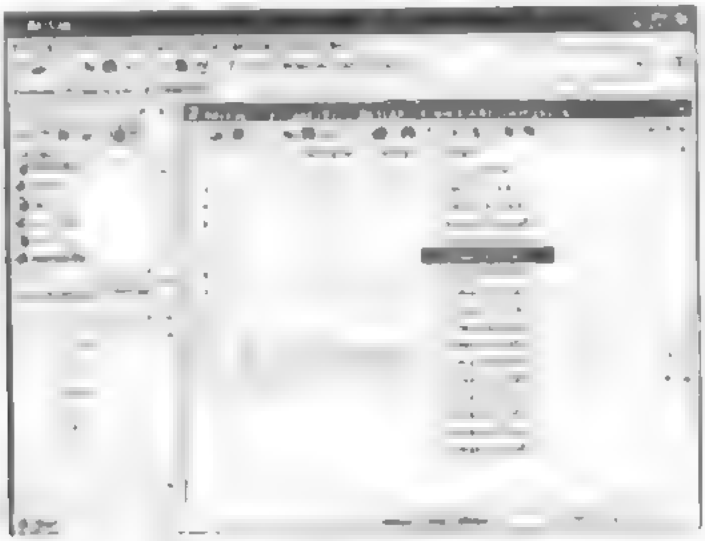


图 11.33 选择对应的程序代码

在 MATLAB 编辑的位置上，编写对应的程序代码，详细代码如下

```
[illegible]
  hObject    handle to X_Comp (see GCBO)
[illegible]
  Hint: get(hObject,'String') returns contents of X_Comp as text
        str2double(get(hObject,'String')) returns contents of X_Comp
[illegible]

set(handles.Draw,'Enable','On');          % Enable Draw_pushbutton after
[illegible]
```

[illegible]

[illegible]

step 5 [illegible] 编写对应的程序代码，编写后的代码如下

```
function X_range_Callback(hObject, eventdata, handles)
% hObject    handle to X_range (see GCBO)
[illegible] MATLAB
% handles    structure with handles and user data (see GUIDATA)
[illegible]
```

```
%         str2double(get(hObject,'String')) returns contents of X_range
as a double

guidata(hObject, handles);
```

上面程序代码的功能十分简单，就是将用户在"X_range"控件中输入的数值传递给对应的程序变量。对于其他的编辑框控件，用户可以添加相同的程序代码。

step 6 编写"Draw"按钮的回调函数。在MATLAB的命令窗口中，选择"Draw_Callback"选项，然后编写对应的程序代码，编写后的代码如下

```
function Draw_Callback(hObject, eventdata, handles)
% hObject    handle to Draw (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
cla;          % 清除图形窗口中的图形
x_range = str2num(get(handles.X_range,'String'));
y_range = str2num(get(handles.Y_range,'String'));
z_range = str2num(get(handles.Z_range,'String'));
x_comp = get(handles.X_Comp,'String');
y_comp = get(handles.Y_Comp,'String');
z_comp = get(handles.Z_Comp,'String');
%计算图形二维方向上的刻度范围
xmin = x_range(1);
xmax = x_range(2);
ymin = y_range(1);
ymax = y_range(2);
zmin = z_range(1);
zmax = z_range(2);
%计算三维方向上的刻度间隔
dx = (xmax-xmin)/4;
dy = (ymax-ymin)/4;
dz = (zmax-zmin)/4;
%判断间隔是否为0
if dx ~= 0
    xm = xmin:dx:xmax;
else
    xm = zeros(1,5);
end
    if dy ~= 0
    ym = ymin:dy:ymax;
else
    ym = zeros(1,5);
end
    if dz ~= 0
    zm = zmin:dz:zmax;
else
    zm = zeros(1,5);
end
%创建三维数据网格数据点
[meshx,meshy,meshz]=meshgrid(xm,ym,zm);
if x_comp ~= '0'
x_fun = inline(x_comp,'x','y','z');                    % 创建x向函数的内联函数
    xc = x_fun(meshx,meshy,meshz);
else
```

```
    xc = zeros(5,5,5);
end
if y_comp ~= '0'
    y_fun = inline(y_comp,'x','y','z');                    %创建y向函数的内联函数
    yc = y_fun(meshx,meshy,meshz);
else
    yc = zeros(5,5,5);
end
if z_comp ~= '0'
    z_fun = inline(z_comp,'x','y','z');                    %创建z向函数的内联函数
    zc = z_fun(meshx,meshy,meshz);
end
%设置绘制矢量的坐标绝数组
%将坐标轴的句柄存入MATLAB的工作空间
handles.field_axes_handle = gca;
quiver3(meshx,meshy,meshz,xc,yc,zc)
axis vis3d
%建立坐标轴名称
xlabel('x')
%启用网格线
%计算曲度数组
[curlx,curly,curlz]=curl(meshx,meshy,meshz,xc,yc,zc);
%设置绘制矢量坐标轴属性，并将坐标轴的句柄返回工作空间
quiver3(meshx,meshy,meshz,curlx,curly,curlz)
axis vis3d
grid on
%计算数值散度
div = divergence(meshx,meshy,meshz,xc,yc,zc);
%将坐标轴属性返回工作空间
handles.position = [meshx(:),meshy(:),meshz(:)];
handles.divergence = div(:);
```

MATLAB的workspace中。

step 7 [illegible]

```
% --- Executes on button press in Output.
function Output_Callback(hObject, eventdata, handles)
% hObject    handle to Output (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
set(handles.Close,'Enable','On');      % enable close_pushbutton after
[illegible]

% --- Executes on button press in Close.
function Close_Callback(hObject, eventdata, handles)
% hObject    handle to Close (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[illegible]
```

11.3.5 运行 GUI 对象

经过上面小节的步骤，

step 1 [illegible] MATLAB 工作目录，输入 "[illegible]"，[illegible] 或 [illegible] 命令，[illegible] "Draw" 按钮，[illegible]，如图 11.34 所示。

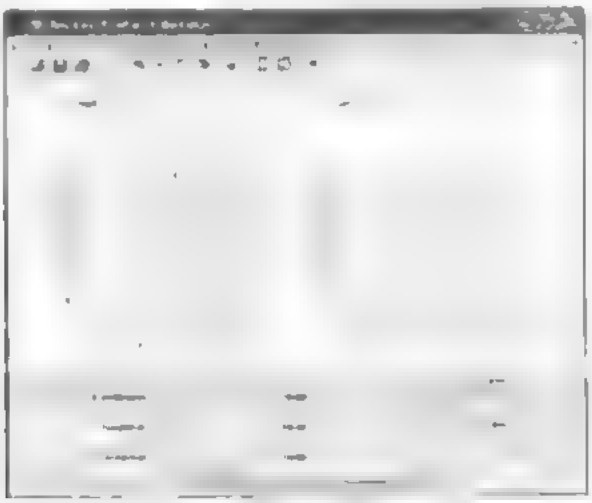


图 11.34 运行 GUI 对象

[illegible] "Output" [illegible]

step 2 [illegible] 的 "Close" 按钮[illegible]，如图 11.35 所示。

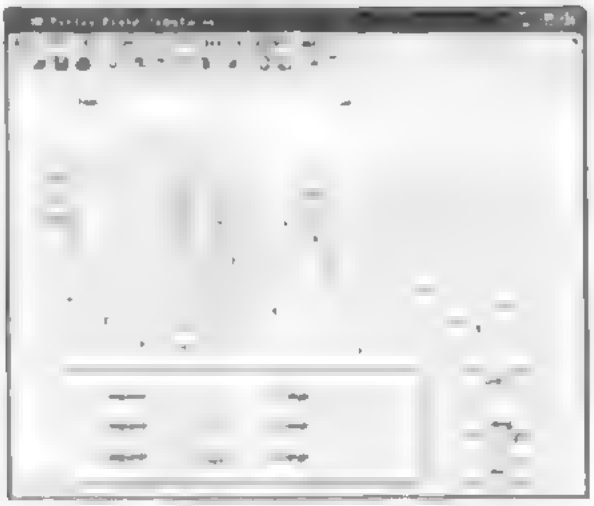


图 11.35 退出应用程序

[illegible]时退出相应的应用程序。

[illegible]

11.3.6 GUIDE 创建 GUI 的注意事项

[illegible]对象为例。

[illegible]开"Align Objects"对话框，如图 11.36 所示。

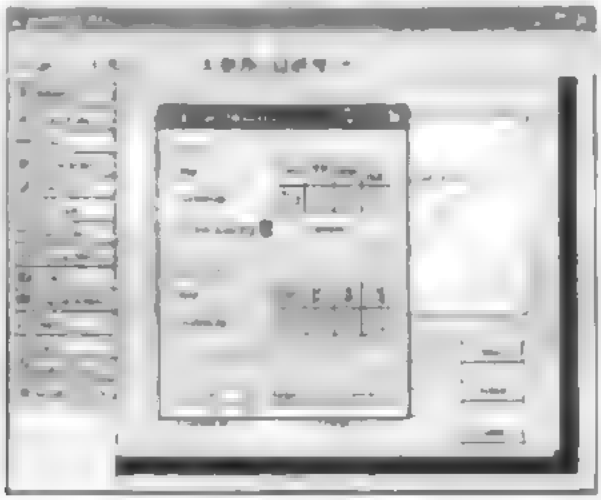


图 11.36 排列对象对话框

[illegible]

[illegible]

打开“Grid and Rulers”对话框，如图 11.37 所示。

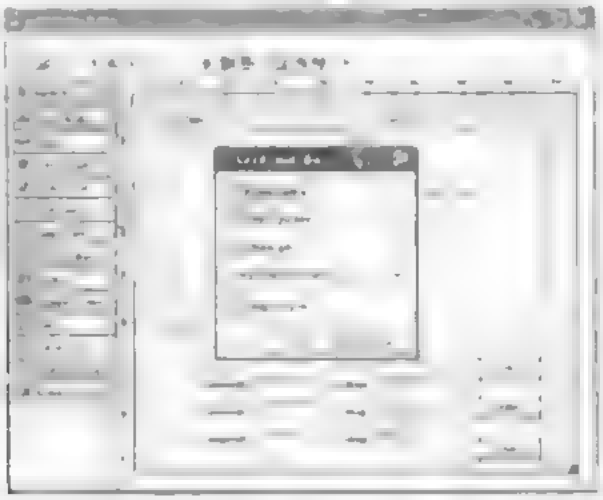


图 11.37 网格和标线对话框

[illegible]

[illegible]

对话框，如图 11.38 所示。

[illegible]

[illegible]

图 11.38 对象浏览器

[illegible]

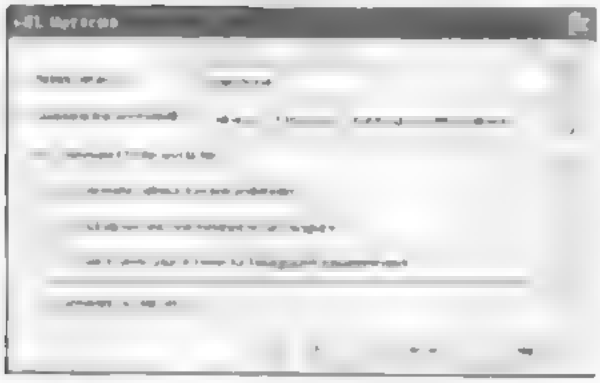


图 11.39 GUI 应用程序的选项对话框

[illegible] GUI 的各种属性，下面详细介绍对话框各个选项的具体含义。

11.3.7 重画行为（Resize behavior）

[illegible]

- Non-resizable：[illegible]
- Proportional：[illegible] 哪来讲，该选项是最佳的选择。
- User-specified：[illegible] 小和位置。

11.3.8 命令行访问（Command-Line Accessibility）

[illegible] 来访问该图形句柄。

在MATLAB中，可以为GUI设置如下三种命令访问权限。

- **off**：禁止命令对GUI图形窗口的访问。在这种情况下，GUI图形窗口的图形句柄是隐藏的。为此，应用程序M文件将创建一个对象句柄结构体来保存GUI中的所有用户控件句柄，并将该结构传递给子函数，保证GUI中句柄的准确使用。
- **on**：允许命令行对GUI图形窗口进行访问。
- **User-specified**：用户可以设置Handle Visibility和IntergerHandle两个图形窗口属性值，决定命令行是否能够访问图形窗口的句柄。Handle Visibility属性决定图形窗口的句柄对访问当前图形窗口的命令行是否可见，如果该属性值为off，则图形窗口的句柄从根对象的子对象列表中删除，因此该图形窗口将不是当前图形窗口，但是该图形窗口的句柄依然有效。IntergerHandle属性决定图形窗口的句柄是一个整数还是浮点数。如果该属性值为off，MATLAB将使用浮点数来代替整数。

生成FIG文件和M文件（Generate FIG-file and M-file）

如果希望通过使用GUIDE创建GUI对象的时候，同时生成FIG文件和应用程序M文件，则可以在该对话框中选中“Generate FIG file and M file”选项。选中该选项后，用户可以对其设置相应的属性选项，下面详细介绍。

- **生成回调函数原型（Generate Callback Function Prototypes）**：当选中该选项后，GUIDE将会在应用程序M文件中为每一个控件添加一个回调函数（需要提醒用户的是，组合框和静态文本控件不包含Callback属性）。用户可以自行添加回调函数的程序代码。
- **同一时间只允许运行一个应用程序实例（GUI Allows Only One Instance to Run（Singleton））**：选中该选项后，MATLAB在一次应用程序运行过程中只有一个GUI实例。如果GUI已经存在，MATLAB将该GUI带到前台，而不会重新创建一个新的GUI图形窗口。如果允许MATLAB显示多个GUI实例，则每一个调用命令都会创建一个新的图形窗口。
- **使用系统背景颜色设置（Using the System Background Colors）**：GUI控件所使用的颜色和计算机系统有关。如果选中该选项，则使图形窗口的背景颜色和用户添加的控件默认背景颜色相互匹配。

仅生成FIG文件（Generate FIG-file only）

如果不希望GUIDE生成对应的应用程序M文件，可以选中该选项。当用户在界面设计编辑器中保存GUI对象时，GUIDE将仅仅创建FIG文件，用户可以使用open命令或者hgload命令来显示该文件。当希望创建一个与应用程序M文件完全不同的实例时，可以选中该选项。

定制标准菜单

在GUI控件对象中，界面菜单（uimenu）是一个重要的组成部分。从句柄图形对象结构中，Uimenu对象的结构体系以Figure图形窗口为父对象，和Axes坐标轴、Unicontrol界面控件为平等级别的组件。

在MATLAB中，可以根据需要在GUI对象中包建标准菜单，自行设置菜单或者创建现场菜单等。同时，可以设置菜单控件的各种属性，例如添加快捷键、设置对应的回调函数等。因此，在GUI对象中，可以设置菜单控件来完成多种功能。对于比较简单的GUI对象，可以根据需要定制MATLAB图形

[illegible]

例11.3 创建一个简单的GUI对象，根据需要定制自定菜单。

Step 1 [illegible]

```
>> Handle_figure=figure;
```

Step 2 [illegible]

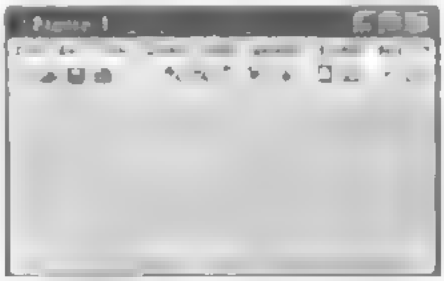


图11.46 创建默认的窗口

[illegible]产生对应的下拉菜单（Pull-Down Menu）。

[illegible]

Step 3 [illegible]如下

```
>> set(Handle_figure,'MenuBar'
[ none | { figure} ]
```

[illegible]

Step 4 [illegible]命令窗口中输入下面的代码：

```
>> set(Handle_figure,'MenuBar',menubar);
```

Step 5 [illegible]

[illegible]

图 11.41 隐藏菜单选项后的图形窗口

Step 6 [illegible]

```
>> set(Handle_figure,'MenuBar','figure')
```

Step 7 [illegible]

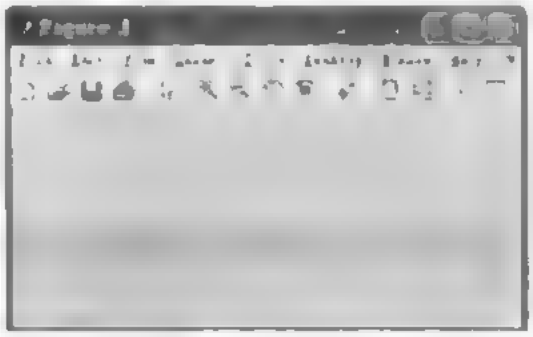


图 11.42 恢复菜单后的图形窗口

11.5 使用 GUIDE 创建自定义菜单

[illegible]小节详细介绍。

11.5.1 创建图形界面

例 11.4 [illegible]如图 11.43 所示。

[illegible]

Step 1 [illegible]对话框，如图 11.44 所示。

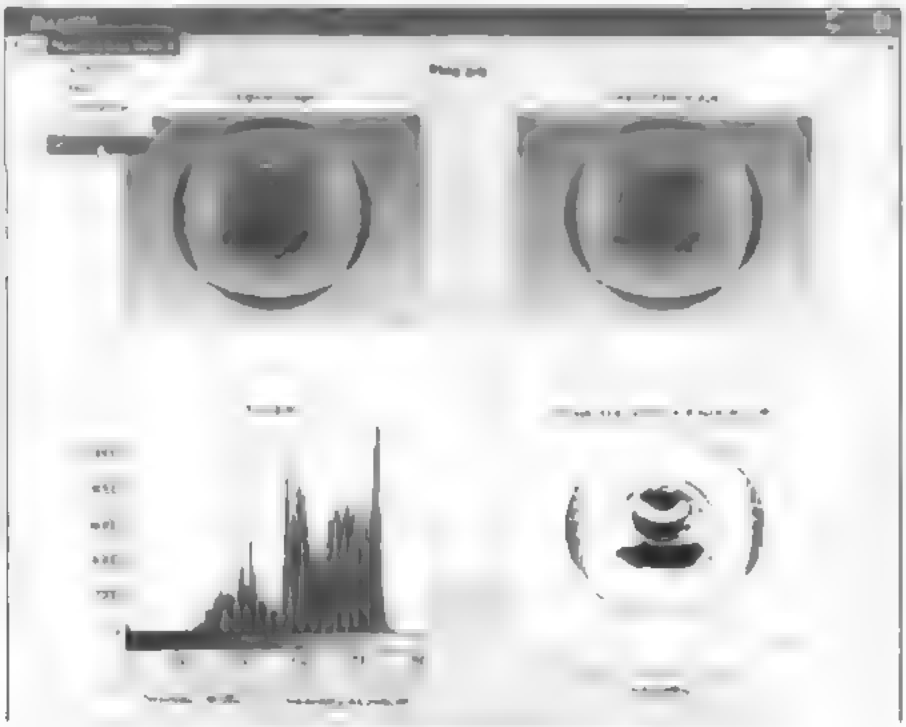


图 11.43　生成后的GUI界面

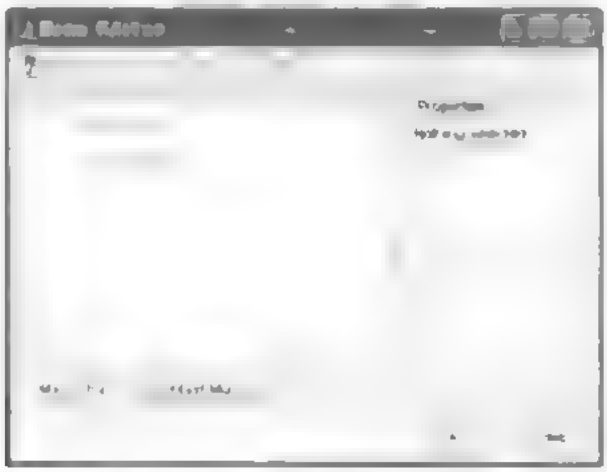


图 11.44　菜单编辑器的外观

[illegible]

[illegible]

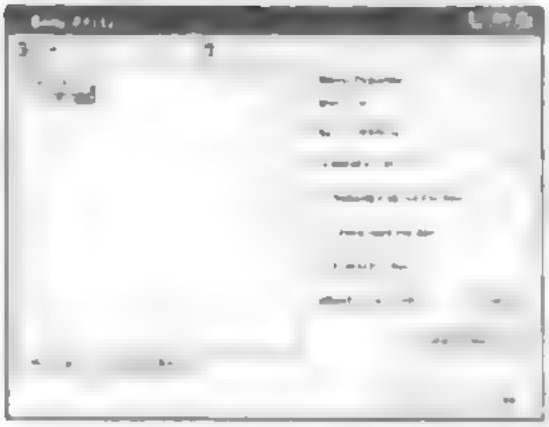


图 11.45 创建一个菜单选项

[illegible]

[illegible]

Step 3 [illegible]

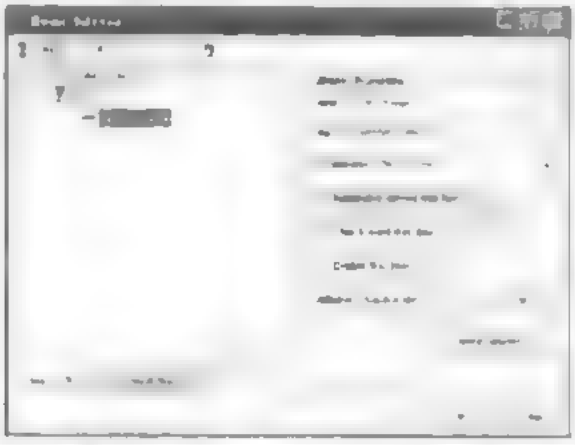


图 11.46 添加第一个子菜单选项

[illegible]

Step 4 [illegible]

其他子菜单选项，如图 11.47 所示。

上面步骤中添加的两个子菜单选项如下。

- **Label:** Save Image，**Tag:** Save。
- **Label:** Close，**Tag:** CloseMenuItem。

图 11.47　添加其他子菜单选项

[illegible]

[illegible]

Step 3 [illegible] 11.48 所示。

图 11.48　前面步骤设置的结果

[illegible]

Step 4 [illegible] 单击打开“Menu Editor”对话框，如图 11.49 所示。

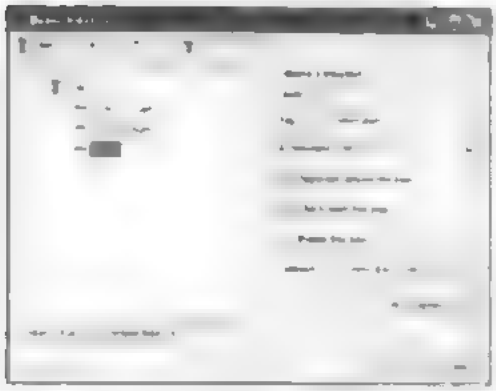


图 11.49 重新打开“Menu Editor”对话框

[illegible]

Step 7 [illegible] 11.50 所示。

图 11.50 添加新的菜单选项

[illegible]

Step 8 [illegible]

[illegible]

11.5.2 使用图形界面工具栏

继续上面小节的步骤。

图 11.51　容易菜单设置的结果

STEP 3　[illegible]

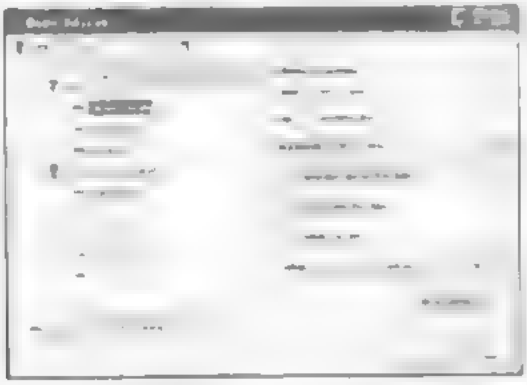


图 11.52　菜单设置对话框

STEP 4　[illegible]
修改各种菜单条目的属性。具体的工具栏如图 11.53 所示。

图 [illegible]

图 11.53　工具栏按钮

将上面的工具栏按钮依次命名为 1～8。下面详细介绍各按钮的功能：

- **按钮 1**：[illegible] 单选项。
- **按钮 2**：[illegible]
- **按钮 3**：[illegible] 时，才可以使用该按钮。
- **按钮 4**：[illegible] 中。当用户单击该按钮后，子菜单将会被移到上层，变成菜单栏菜单。

- 按钮5：[illegible]
- 按钮6：[illegible]菜单选项，当用户单击该按钮后，子菜单选项会向上移动一个单位。
- 按钮7：[illegible]子菜单选项，当用户单击该按钮后，子菜单选项会向下移动一个单位。
- 按钮8：[illegible]菜单选项。

[illegible]

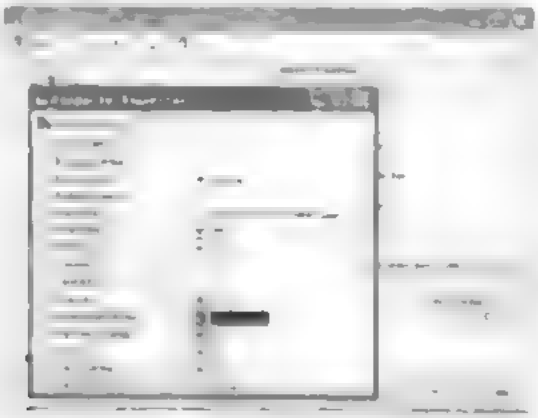


图 11.54 设置菜单选项的属性

[illegible]

11.5.3 添加图形界面的控件

接续上面小节的步骤

Step 1 添加控件。根据本小节的要求，添加后的结果如图 11.55 所示。

Step 2 [illegible]件的类型、功能和重要属性。

- [illegible]显示各种图像。
- [illegible]地调整滚动条的参数数值。
- [illegible]
- [illegible]"Umbralizer"，"Visible"属性为"Off"，功能要实现图形转换功能。

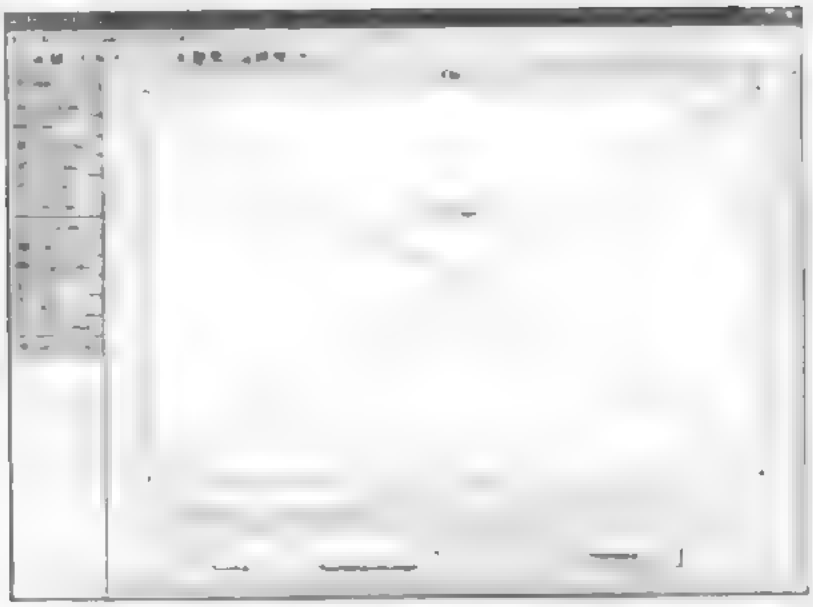


图 11.55　添加图形控件

Step 3

图 11.56　查看 GUI 的对象结构

11.5.4　添加"File"菜单的回调函数

接续上面小节的步骤。

Step 1

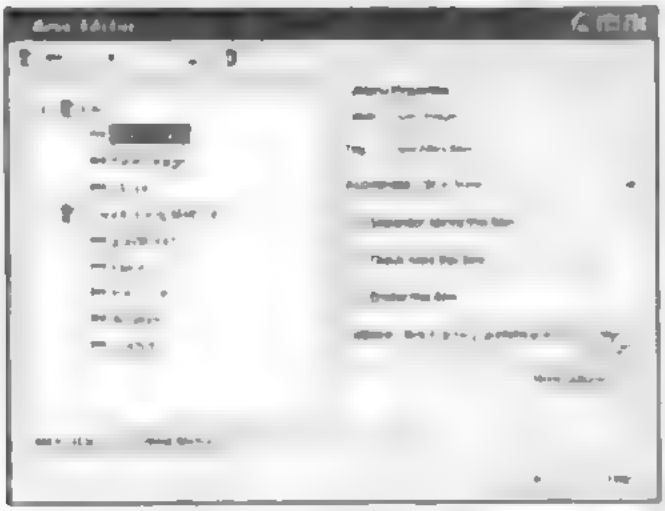


图 11.57 编写“Open Image”的回调函数

step 2 [illegible]

```
function OpenMenuItem_Callback(hObject, eventdata, handles)
% 定义全局变量
global [illegible]

% 将 save 菜单选项设置为不可用
[illegible]
set(handles.TipoSegmentacion,'enable','off')

% 打开图像文件
[NameArchivo,PathArchivo] = uigetfile({'*.jpg;*.tif;*.gif;*.bmp;*.png;
[illegible]

% 如果选择图像，则执行以下操作，[illegible]
if ~isequal(NameArchivo, 0)
    [illegible]
    CargarImagen(handles)
    [illegible]
    set(handles.TipoSegmentacion,'enable','on')
    set(handles.DefinidoUsuario,'checked','on')
end
```

[illegible]

step 3 [illegible]

```
function Save_Callback(hObject, eventdata, handles)
% 定义全局变量
global ImagenUmbral
% 如果在单形界面中没有加载图形文件，则显示错误信息
[illegible]
```

```
end
%显示保存文件的对话框
[filename,pathname] = uiputfile('*.jpg', '*.bmp', '*.png', '*.tif', '*.gif',
    '*.hdf', '*.pcx', '*.xwd', '*.ico', '*.cur', '*.tga',
    '*.pnm', '*.pgm', '*.ppm','Save file name');
%[filename,pathname] = uiputfile('*.jpg','Save file name');
if isequal(filename,0) | isequal(pathname,0)
   errordlg('Saving canceled','Threshold GUI'); error('Saving canceled')
else
    try
        imwrite(Imagen1mbral,[pathname,filename]);
    catch
        [filename,pathname] = uiputfile('*.jpg','Save file name');
        imwrite(Imagen1mbral,[pathname,filename]);
    end
end
end %%%
```

在上面一段代码中，使用了[illegible]保存文件的代码。

Step 6 [illegible]文件的代码如下。

```
[illegible]
end
%%%
```

[illegible]

11.5.5 添加“Thresholding Method”菜单的回调函数

继续上面小节的步骤。

Step 1 [illegible]在右侧的“View”，添加相应的函数代码，如图11.58所示。

图 11.58 添加菜单选项的程序代码

step 2

```
function graythresh_Callback(hObject, eventdata, handles)

% 将其他菜单选项的属性设置为不选中

% 将 "graythresh" 菜单选项的属性设置选中

% 调用graythresh灰度转换函数
Umbral = graythresh(ImagenGris)*255;

global ImagenGris
% 将所有菜单选项的属性设置中不选中

% 将 "kapur" 菜单选项的属性设置选中
set(handles.kapur,'checked','on')
% 调用Kapur1 灰度转换函数
Umbral = Kapur1(ImagenGris)*255;
GraficarHistoUmbral(Umbral,handles,'all')

function triangular_Callback(hObject, eventdata, handles)

% 将所有菜单选项的属性设置为不选中

% 将 "triangular" 菜单选项的属性设置为选中
set(handles.triangular,'checked','on')
% 调用triangular 灰度转换函数
Umbrales = Triangular1(ImagenGris)*255;

global ImagenGris
% 将所有菜单选项的属性设置为不选中
NoChecked(handles)
```

```
%将 "iterativo" 菜单选项的属性设置为选中
set(handles.iterativo,'checked','on')
%调用iterativo图形转换函数
Umbral = Iterativo1(ImagenGris)*255;
GraficarHistoUmbral(Umbral,handles,'all')
% ----------------------------------------------------------------------
function DefinidoUsuario_Callback(hObject, eventdata, handles)
% ----------------------------------------------------------------------
%将所有菜单选项的属性设置为不选中
NoChecked(handles)
%将 "DefinidoUsuario" 菜单选项的属性设置为选中
set(handles.DefinidoUsuario,'checked','on')
set(handles.Umbralizar,'enable','on')
set(handles.Umbralizar,'string','Thresholding')
if get(handles.ActualAutomatico,'value')==1
    set(handles.Umbralizar,'enable','off')
end %if
```

step 3 查看图形转换函数的代码。在上面的程序代码中，对不同的菜单选项，调用不同的图形转换函数，这些函数都是用户自行定义的函数，对应的函数代码如下：

```
%=====================================================================
function umbral = Kapur1(Imagen)
%=====================================================================
%检测图像的属性
if ~Esgray(Imagen)
   uiwait(msgbox('La Imagen no se encuentra en escala de grises','Error',
'modal'))
    umbral = 0;
else
[fil col] = size(Imagen);
%计算图像的像素数值
Pixeles = fil * col;
%计算图像的直方图像素值
    hl = imhist(Imagen);
Pi = hl/Pixeles;
Pt = zeros(256,1);
    Pt(1) = Pi(1);
    for i = 2:256
        Pt(i)=Pt(i-1)+Pi(i);
    end
%创建循环进行图像转换
    Hb = zeros(1,256);                                  Hw = zeros(1,256);
    for i = 1:256
        if Pt(i) > 0
            for j = 1 : i
                if Pi(j) > 0
                    Hb(i) = Hb(i) + ((Pi(j) / Pt(i)) * log(Pi(j) / Pt(i)));
                end
            end
        end
    end
    for i = 1:256
        if (1-Pt(i)) > 0
            for j = i + 1 : 256
```

```
                if Pi(j) > 0
                    Hw(i) = Hw(i) + ((Pi(j) / (1-Pt(i))) * log(Pi(j) / (1-Pt(i))));
                end
            end
        end
    end
    Hb =  Hb;
    Hw = -Hw;
    H = Hb + Hw;
    [a, b] = max(H(:));
    umbral = b-1;
    umbral = umbral/255;
end

%==========================================================================
function Umbral = Triangular1(Imagen);
%==========================================================================
%检测图像的属性
if ~Esgray(Imagen)                                          %Validar si una
Imagen se encuentra en Escala de Grises o en color
    uitwait(msgbox('La imagen no se encuentra en escala de grises',
'Error','modal'))
    Umbral = 0;
else
z = 0;
%返回图像的直方图数据
H = imhist(Imagen);
%调用 Promedio 函数，返回新的图像句柄
H = Promedio(H);
%调用 Derivada 函数，返回 dh 数值
dh = Derivada(H);
%调用 MaxiMini 函数，返回 dh 的最小值和最大值
[maxim minim] = MaxiMini(dh);
%使用循环进行图像转换
    for i = 1 : length(maxim)-1;
        clear D, k = 0;
        x1 = maxim(i); y1 = H(maxim(i));
        x2 = maxim(i + 1); y2 = H(maxim(i + 1));
        M = (y2 - y1) / (x2 - x1);
        if (x2   1) - (x1 + 1) > 0
            for j = x1 : x2;
                Px = j;
                Py = H(j);
                k = k + 1;
                D(k,1) = sqrt(((Px - x1) * M - Py + y1)^2 / (M^2 + 1));
                D(k,2) = j;
            end
            [Pu p] = max(D(:,1));
            if Pu > 0
                z = z + 1;
                Umbral(z) = D(p,2);
            end
        end
    end
    Umbral = Repetidos(Umbral,10)/255;
end
```

```
% Promedio子函数的子程序代码
function x = Promedio(h);
n = 5;
%创建零值矩阵
g = zeros(256 + 2 * round(n / 2 - 1), 1);
%将图像的直方图数据添加到矩阵g中
g(1 + round(n / 2 - 1) : length(g) - round(n / 2 - 1)) = h;
%利用循环实现图像转换
for i = 1 + round(n / 2 - 1) : length(g)-round(n / 2 - 1);
    sum = 0;
    for j = i   round(n / 2 - 1) : i + round(n / 2 - 1)
        sum = sum + g(j);
    end
    x(i - round(n / 2 - 1), 1) = round(sum / n);
end
% Derivada子函数的子程序代码
function dx = Derivada(h);
%创建零值矩阵
dx=zeros(256,1);
%通过函数公式来返回转换的数值
for i=3:254;
    dx(i)= (h(i-2) - 8*(h(i-1)) + 8*(h(i+1)) - h(i+2))/12;
end
%MaxiMini子函数程序代码
function [Maxi, Mini] = MaxiMini(d);
j = 1; y = 1;
for i = 2 : 256;
    if (d(i - 1) < 0 && d(i) >= 0)
        Mini(j) = i;
        j = j + 1;
    elseif (d(i - 1) >= 0 && d(i) < 0)
        Maxi(y) = i;
        y = y + 1;
    end
end

function Vector = Repetidos(Arreglo, D);
N_Ind = length(Arreglo);
for i = 1 : N_Ind - 1;
    if Arreglo(i) > 0
        for j = i + 1 : N_Ind;
            if Arreglo(i) == Arreglo(j) || (abs(Arreglo(i)-Arreglo(j))<D)
                Arreglo_Nuevo(j) = 0;
            else
                Arreglo_Nuevo(j) = Arreglo(j);
            end
        end
    end
end
[i j Vector] = find(Arreglo_Nuevo);

function Umbral = Iterativo1(Imagen)
%==========================================================================
%检测图像文件的属性
if ~Esgray(Imagen)
    uitwait(msgbox('La imagen no se encuentra en escala de grises',
```

```
    Umbral = 0;
end
%返回直方图数据的向量
Histograma = imhist(Imagen);
%返回直方图数据中的非零数值的下标
Grises = find(Histograma);
%返回最大下标和最小下标
    Maximo = max(Grises);
    Minimo = min(Grises);
    Umbral = Minimo + (Maximo - Minimo) / 2;
Umbralp = 0;
%使用迭代进行图像转换
    while (abs(Umbral - Umbralp) > 1)
        Mayores = find(Imagen > Umbral);
        Menores = find(Imagen <= Umbral);
        m1 = mean(Imagen(Mayores));
        m2 = mean(Imagen(Menores));
        Umbralp = Umbral;
        Umbral = round((m1+m2)/2);
    end
    Umbral = Umbral/255;
end
%-----------------------------------------------------------------------
function y = Esgray(x)
%-----------------------------------------------------------------------
y = ndims(x)==2 && ~isempty(x);
%
  y = false;
%
  % 选取图像的最小范围进行检测
  [m,n] = size(x);
  chunk = x(1:min(m,10),1:min(n,10));
  y = min(chunk(:))>=0 && max(chunk(:))<=1;
  % 如果 chunk 是灰度图像，则检查整个图像
  if y
      y = min(x(:))>=0 && max(x(:))<=1;
  end
end
```

在上面的程序代码中，主要使用了 kappa、[illegible] 和 [illegible] 三个函数来检查输入参数的合法性。kappa 函数主要用于检查输入参数的个数，[illegible]

Step 4 查看 graythresh 函数的代码，[illegible]。在上面的程序代码中，[illegible] graythresh 函数。这个函数是 MATLAB 的内置函数，可以通过命令

```
function level = graythresh(I)
% 需要一个输入参数
% 检测输入参数的数值
checknargin(1,1,nargin,mfilename);
[illegible],
1);
```

```
if ~isempty(I)
  % 将所有的数值数组转换为单列数据
  % 转换为uint8的数据格式，有助于图像转换的运算
  I = im2uint8(I(:))
  num_bins = 256;
  counts = imhist(I,num_bins);
  % 下面的参数名称和图像转换公式中的参数名称相同
  p = counts / sum(counts);
  [illegible]
  mu = cumsum(p .* (1:num_bins)');
  mu_t = mu(end);
    previous_state = warning('off', 'MATLAB:divideByZero');
  sigma_b_squared = (mu_t * omega - mu).^2 ./ (omega .* (1 - omega));
  [illegible]
  [illegible]
  if isfinite(maxval)
    idx = mean(find(sigma_b_squared == maxval));

    % 单位化转换类位的范围
    level = (idx - 1) / (num_bins - 1);
  else
    level = 0.0;
  end
else
  level = 0.0;
end
```

Step 5 [illegible]

```
function NoChecked(handles)
[illegible]
% 将所有的菜单条选项置为没有选中
[illegible]
set(handles.kapur,'checked','off')
set(handles.triangular,'checked','off')
set(handles.iterativo,'checked','off')
[illegible]
```

[illegible]的菜单选项调用函数。

[illegible]

11.5.6 添加滚动条的回调函数

继续上面小节的步骤

Step 1 [illegible]

[illegible] "Callback" 命令，如图 11.59 所示。

图 11.59 查看滚动条的回调函数

MATLAB 会[illegible]，[illegible]。

```
% ---- 当滚动条运动时将调用下面的程序代码
%-----------------------------------------------------------------------
function slider1_Callback(hObject, eventdata, handles)

set(handles.Intensidad,'visible','on')
[illegible]
% 调用GraficarHistograma函数
GraficarHistograma(Umbral,handles)
[illegible]
% 调用GraficarUmbral函数
[illegible]
end %if
NoChecked(handles);set(handles.DefinidoUsuario,'checked','on')
```

step 2 [illegible]，如下

```
%-----------------------------------------------------------------------
function GraficarUmbral(Umbral,handles)
%-----------------------------------------------------------------------
global ImagenGris ImagenUmbral
% 将像素转换为二元制影像
imgUmbral=im2bw(ImagenGris,Umbral/255);
[illegible]
% 显示图形
imshow(imgUmbral)
% 添加图形标题
title([ 'Image segmented in threshold = ',int2str(Umbral)])
[illegible]
```

```
%==========================================================================
function GraficarHistograma(Umbral,handles)
%==========================================================================
global ImagenGris
persistent H
set(handles.Intensidad,'String',Umbral)
%绘制图形的直方图
subplot(223);imhist(ImagenGris);title('Histogram')
Escala=axis;
if ishandle(H)==1,delete(H),end%if
H=line([Umbral Umbral],[Escala(3:4)],'color','r');
```

添加其他控件的回调函数

延续上面小节的步骤。

step 1 添加"ActualAutomatico"复选框的回调函数。在GUIDE图形界面中选中"slider1"对象，然后单击右键，在弹出的快捷菜单中选择"View Callbacks"→"Callback"命令，然后在M文件编辑器的对应位置编写代码如下：

```
% --- 当用户单击 ActualAutomatico按钮后，触发该程序代码
%--------------------------------------------------------------------------
function ActualAutomatico_Callback(hObject, eventdata, handles)
%--------------------------------------------------------------------------
%设置Umbralizar对象句柄的属性
set(handles.Umbralizar,'enable','on')
set(handles.Umbralizar,'string','Thresholding')
set(handles.Umbralizar,'enable','off')
if get(hObject,'Value')==1
    set(handles.Umbralizar,'enable','off')
else
    set(handles.Umbralizar,'enable','on')
end %if
NoChecked(handles);set(handles.DefinidoUsuario,'checked','on')
```

step 2 添加"Umbralizar"按钮的回调函数。在GUIDE图形界面中选中"Umbralizar"对象，单击右键，在弹出的快捷菜单中选择"View Callbacks"→"Callback"命令，然后在M文件编辑器的对应位置编写代码如下：

```
% ---当用户单击 Umbralizar按钮时触发下面的程序代码
%--------------------------------------------------------------------------
function Umbralizar_Callback(hObject, eventdata, handles)
%--------------------------------------------------------------------------
if isequal(get(handles.triangular,'checked'),'off')==1
Umbral=get(handles.slider1,'value');
%调用GraficarUmbral函数
GraficarUmbral(Umbral,handles);
%确定DefinidoUsuario对象的句柄被选中
    NoChecked(handles);set(handles.DefinidoUsuario,'checked','on')
else
%获取对象句柄的数值
    UmbralActual=get(handles.slider1,'value');
    Umbrales=get(handles.triangular,'userdata');
Pos=find(Umbrales==UmbralActual);
```

```
%返回错误信息
    if isempty(Pos),msgbox('Error de Ejecuci');return,end
if Pos==length(Umbrales),Pos=0;,end %if
%调用GraficarHistoUmbral函数
    GraficarHistoUmbral(Umbrales(Pos+1),handles,'')
end %if
```

step 3 查看图像转换函数的代码。在上面的程序代码中，调用了另外一种图像转换函数GraficarHistoUmbral，其详细的函数代码如下：

```
%==========================================================================
function GraficarHistoUmbral(Umbral,handles,Tipo)
%==========================================================================
%调用GraficarHistograma和GraficarUmbra函数
GraficarHistograma(Umbral,handles)
GraficarUmbral(Umbral,handles)
set(handles.slider1,'value',Umbral)
%设置对象的属性数值
switch Tipo
    case 'all' %GRAYTHRESH ITERATIVO KAPUR
        set(handles.ActualAutomatico,'value',0)
        set(handles.Umbralizar,'enable','on')
        set(handles.Umbralizar,'string','Thresholding')
        set(handles.Umbralizar,'enable','off')
    case 'trian' %TRIANGULAR
        set(handles.ActualAutomatico,'value',0)
        set(handles.Umbralizar,'enable','on')
        set(handles.Umbralizar,'string','next Threshold >>')
end %switch
```

编写主调函数

延续上面小节的步骤。

编写"CargarImagen"函数。其对应的程序代码如下：

```
%==========================================================================
function CargarImagen(handles)
%==========================================================================
global Archivo NameArchivo ImagenGris
%绘制原始图形
subplot(221);img=imread(Archivo);imshow(img);title('Original Image');
axis off
%绘制灰度图形
try
    if Esgray(img)==0
        imggris=rgb2gray(img);subplot(222);imshow(imggris);title('Gray
Scale Image')
    else
         imggris=img;subplot(222);imshow(imggris);title('Gray Scale
Image')
    end %if
catch
    errordlg('Error during image processing','Threshold GUI');error
('Error during image processing')
```

```
end stry
%绘制导像信息的直方图
[illegible]
[illegible]
[illegible]
%设置控件的属性
set(handles.slider1,'visible','on')
set(handles.slider1,'Min',Min)
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
set(handles.Archivo,'visible','on')
set(handles.Archivo,'string',NameArchivo)
GraficarUmbral(Min,handles)
```

11.5.9 运行GUI

完成上面小节的步骤，

step 1 [illegible] 步骤中的GUI对象，如图11.60所示。

图11.60 打开GUI对象

[illegible]

step 2 [illegible] 11.61所示。

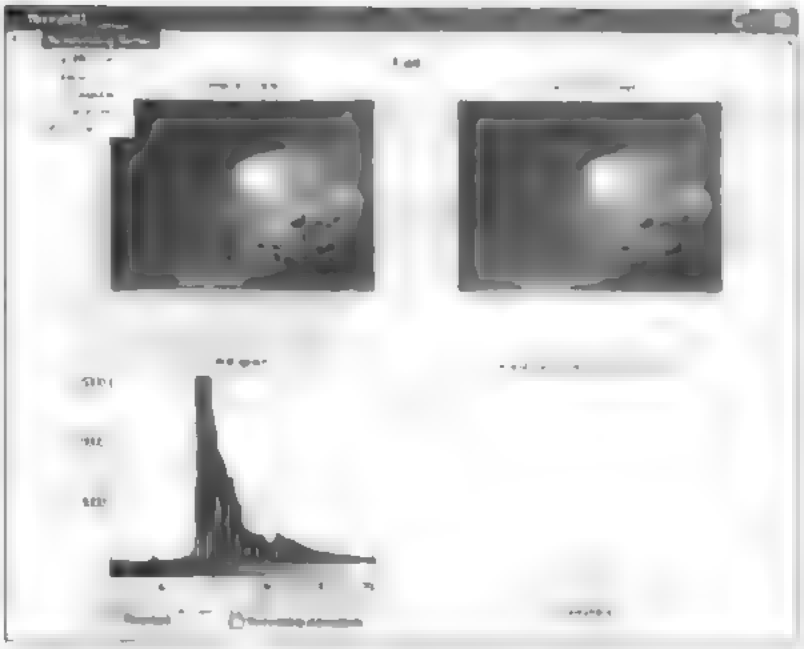


图 11.61 运行整个程序

从上面的图形界面中可以看出，本程序共有三个按钮，[illegible] Matlab 的数学函数全部载入，用户可以选择自己的图像转换方法。

step 3 选择图像转换方法。单击“Hamming”按钮，[illegible]，单击“Hamming”函数，[illegible]

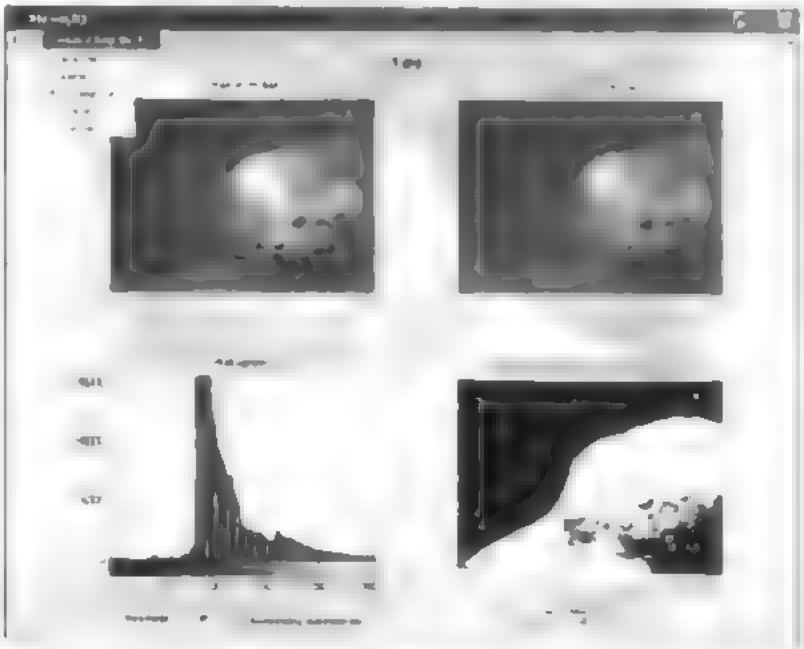


图 11.62 进行图像转换

单击“Hamming”按钮后，图像转换的结果[illegible]，[illegible]，[illegible]“Hamming”[illegible]，[illegible]图像转换的结果。

STEP 4 [illegible]

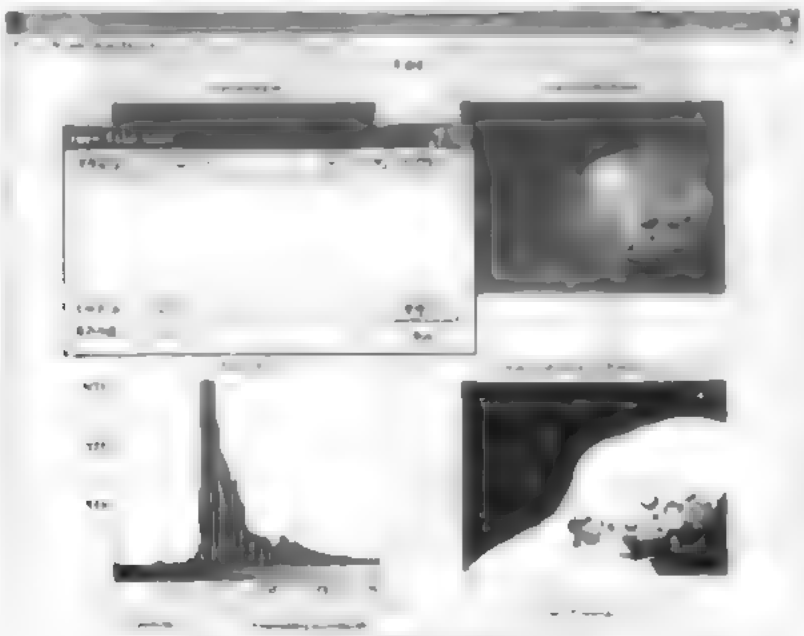


图 11.63 保存图形转换文件

[illegible]，打开 "Save file name" 对话框，用户它以选择要[illegible]

STEP 5 [illegible]

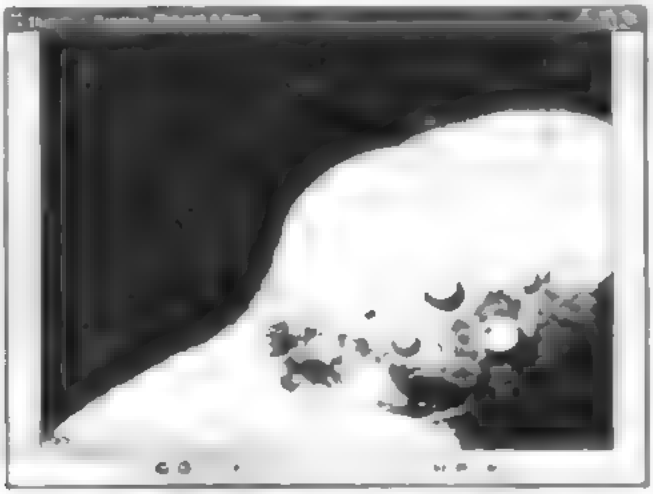


图 11.64 保存的图形文件

[illegible]件转换的图形文件。

STEP 6 [illegible]

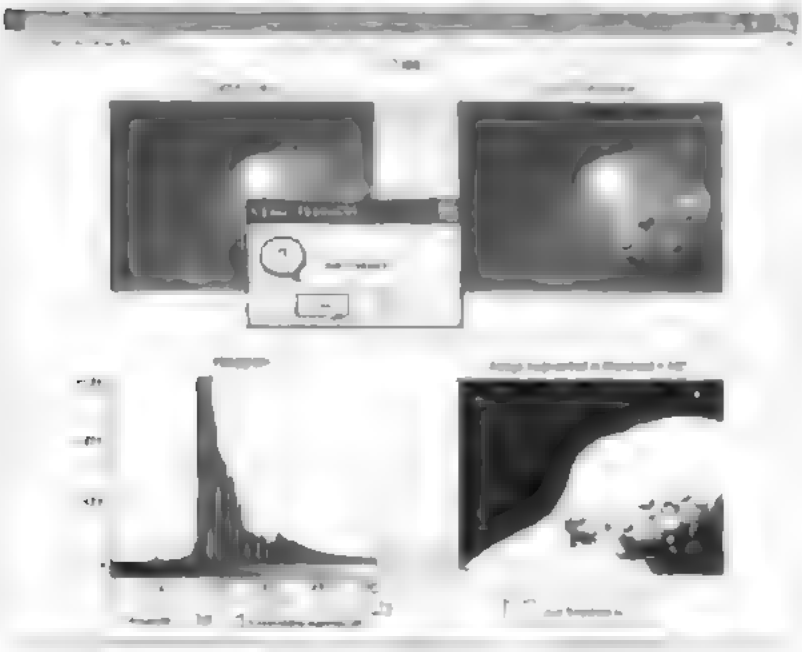


图 11.65 退出 GUI 应用程序

[illegible]

11.6 使用 M 文件创建自定义菜单

[illegible]

11.6.1 演示 GUI 的功能

例 11.5 [illegible] 要求演示 GUI 对象的操作过程。

Step 1 [illegible]

[illegible]

Step 2 [illegible]

初始状态。

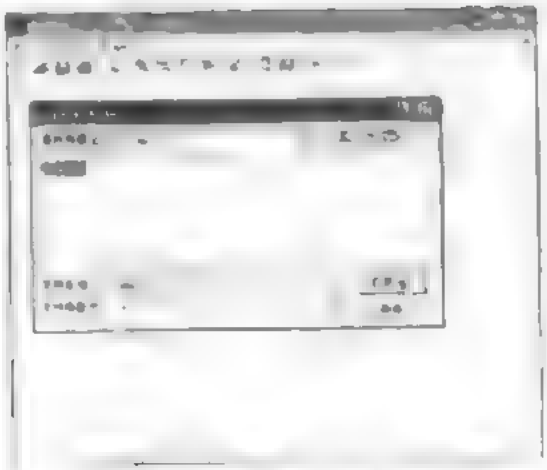


图 11.66　读入数据文件

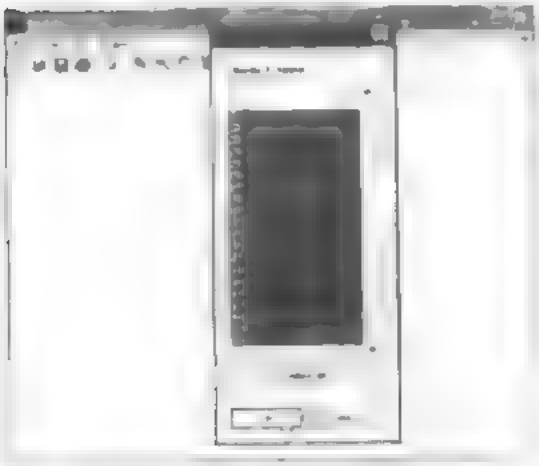


图 11.67　定义图表的数据系列

Step 3 [illegible] 选项，绘制读入的数据系列，如图 11.68 所示。

图 11.68　选择绘图菜单

Step 4 [illegible]

图 11.69 绘制的图形结果

[illegible]标准菜单。

由于每一个菜单选项的作用都是给不同的调用函数，为了[illegible]对象的功能，在本[illegible]，这些每个函数编写一个 M 文件[illegible]，第一个 GUI 对象将是一个包含若干 M 文件的菜单文件。

11.6.2 添加“File”菜单的程序代码

从本小节开始，将详细介绍如何创建上面的 GUI 对象。

STEP 1 [illegible]的，依此，有必要设计菜单选项的层系结构。

- **File 菜单**：[illegible] File、Selected Data、Save as 和 Quit 菜单选项。
- **Options 菜单**：[illegible]的选项，定义各种图表数据系列。
- **Graphs 菜单**：[illegible]表达式，这个菜单是绘图的主要菜单选项命令。

STEP 2 [illegible]的程序代码

```
function getdata (option)
global A opt leg label
%打开外部文件
[fname,pname] = uigetfile('*.xls','Select File');
%创建外部文件的完整路径字符串
dbfile= strcat(pname,fname);
%如果没有打开文件，则退出程序
```

```
if length(dbfile) == 0 return; end
%根据不同选项，读入文件信息
switch option
    case 1
         [A,leg] = readdata(dbfile);
    case 2
         [A,leg] = xlsread(dbfile);
    case 3
         [A,leg] = xlsread(dbfile,-1)
end

%创建元胞数组
if isempty(leg) leg= cellstr( num2str( flipud(rot90([1:size(A,2)])) )
); end
opt.xc= 1;
%根据读入数据系列的列数，决定参数数值
switch size(A,2)
    case 1
         opt.yc= 1;
         opt.ec= 1;
         opt.zc= 1;
    case 2
         opt.yc= 2;
         opt.ec= 2;
         opt.zc= 2;
    otherwise
         opt.yc= [2:size(A,2)-1];
         opt.ec= [fix(size(A,2)/2):size(A,2)];
         opt.zc= [3:size(A,2)];
end
return
```

将上面的程序代码保存为"getdata.m"，该程序代码将是File菜单选项中读取文件的菜单选项对应的程序代码。

step 3 单击MATLAB命令窗口工具栏中的□按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
function savefigas
%保存文件
[fname,pname] = uiputfile('*.fig','Select File');
%创建保存文件的完整路径
dbfile= strcat(pname,fname);
%如果路径为空，则跳出程序代码
if length(dbfile) == 0 return; end
saveas(gcf,dbfile,'fig')
return
```

将上面的程序代码保存为"savefigas.m"，该程序代码将是File菜单选项中"Save as"的菜单选项对应的程序代码。

添加"Options"菜单的程序代码

延续上面小节的步骤。

step 1 单击MATLAB命令窗口工具栏中的□按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
function getcols (col)
global A leg opt
%如果读入的文件中没有数据，显示提示信息，并退出程序
if size(A) == 0    helpdlg('No data. You must read a file first',
'Error'); return, end
%判断选择的数据列
switch col
%如果选择的是x，则将数据读入变量opt的xc维中
    case 'x'
[opt.xc,value]  = listdlg('PromptString','Choose X column',
'SelectionMode','single','ListString',leg);
%如果选择的是y，则将数据读入变量opt的yc维中
    case 'y'
[opt.yc,value]  = listdlg('PromptString','Choose Y column',
'SelectionMode','multiple','ListString',leg);
%如果选择的是z，则将数据读入变量opt的zc维中
    case 'z'
[opt.zc,value]  = listdlg('PromptString','Choose Z column',
'SelectionMode','multiple','ListString',leg);
%如果选择的是e，则将数据读入变量opt的ec维中
    case 'e'
[opt.ec,value]  = listdlg('PromptString','Choose Errors column',
'SelectionMode','multiple','ListString',leg);
end
return
```

将上面的程序代码保存为"getcols.m"，该程序代码将是Options菜单选项中定义数据系列的菜单选项对应的程序代码。

step 2 单击MATLAB命令窗口工具栏中的□按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
function getlabel
global label
%显示提示信息
prompt={'X label','Y label','Z label','Title'};
title= 'Axis Legends';
lines= 1;
resize= 'off';
tmp= inputdlg(prompt,title,lines,struct2cell(label));
fields={'x','y','z','t'};
if size(tmp,1) > 0 label= cell2struct(tmp,fields,1); end
return
```

将上面的程序代码保存为"getlabel.m"，该程序代码将是Options菜单选项中定义数据系列名称的菜单选项对应的程序代码。

step 3 单击MATLAB命令窗口工具栏中的□按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
function window
```

```
global holdon
if (holdon) holdon= 0; else holdon= 1; end
if strcmp(get(gcbo, 'Checked'),'on')
    set(gcbo, 'Checked', 'off');
else
    set(gcbo, 'Checked', 'on');
end
return
```

将上面的程序代码保存为"window.m"，该程序代码将是Options菜单选项中"向图形中添加数据"的菜单选项对应的程序代码。

添加"Graphs"菜单的程序代码

延续上面小节的步骤。

step 1 单击MATLAB命令窗口工具栏中的D按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
function plot2d (type)
global A opt cds holdon leg label figmain
%当数组A 为空，或者数组A的第二维数值小于2
%显示对应的错误信息
if isempty(A) errordlg ('No data present. Sure you read a file yet?',
'Error'); return; end
if size(A,2) < 2 errordlg ('Seems to be only 1 column of data. Need at
least 2','Error'); return; end
%绘制图形
figure (figmain);
if (holdon)   hold on; else   hold off; end
%读入绘图的数据系列
X= A(:,opt.xc);
Y= A(:,opt.yc);
%选择图表类型，绘制对应的图表
switch type
    case 'xyscatter'
         plot(X,Y,'o');
    case 'xyline'
         plot(X,Y,'-o');
    case 'hist'
         hist(A(:,opt.yc(1)),10);
    case 'stem'
         stem(X,Y);
    case 'stairs'
         stairs(X,Y);
    case 'vbarg'
         bar(X,Y,'group');
    case 'vbars'
         bar(X,Y,'stack');
    case 'barerror'
%判断误差线的数据
         if size(Y,2) ~= size(A(:,opt.ec),2)
              msg= errordlg ('Y Data and Error Data have different number
of columns','Error');
              waitfor (msg);
```

```
            [illegible]
        end
%调用barerror函数，绘制误差棒
        [illegible]
    [illegible]
        [illegible]
    case 'hbarg'
        [illegible]
    case 'rose'
        [illegible]
    case 'pie'
        pie(X);
    case 'polar'
        [illegible]
    case 'compass'
        [illegible]
    case 'error'
        if size(Y,2) ~= size(A(:,opt.ec),2)
            msg= errordlg ('Y Data and Error Data have different number
of columns','Error');
            waitfor (msg);
            return;
        end
        multX= [];
        for i= 0:opt.yc(2)-opt.yc(1) multX= [multX,X]; end
        [illegible]
end
[illegible]
legend(leg(opt.yc));
[illegible]
```

[illegible]
项对应的程序代码。

在[illegible]中，[illegible]作代码中，调用了函数[illegible]来实现。

STEP 3 [illegible]的程序代码

```
[illegible]
%判断绘图数组的大小
if mean([ size(X,1),size(Y,1),size(E,1)]) ~= length(X)   error ('Input
vectors are of different lengths'); return; end
if size(Y,2) ~= size(E,2) error ('Data and Error vectors have different
number of columns'); return; end
%设定绘图颜色数组
colors= [ 'k';'w';'r';'g';'b';'c';'m';'y'];
hold on
ncol= size(Y,2);
off= [ fix(-ncol/2):fix(ncol/2)];
[illegible]
if ~mod(ncol,2) off= [off(1:ceil(length(off)/2)-1), off(1+ ceil(length
```

```
(off)/2):length(off))}; end
for h= 1:ncol
    Xtmp= X(:,1)+ off(h)*(realwidth/2)- sign(off(h))*(~mod(ncol,2)
*realwidth/4);
%绘制直方图
    bar(Xtmp,Y(:,h),width/(2*ncol),colors(mod(h,1+length(colors))));
%绘制误差线
    errorbar(Xtmp,Y(:,h),E(:,h),'LineStyle','none','Color',color);
end
hold off
return
```

将上面的程序代码保存为"barerror.m"，该程序代码将是Graphs菜单选项中"2D"的菜单选项下"barerror"菜单选项对应的程序代码。

step 3 单击MATLAB命令窗口工具栏中的D按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
function plot3d (type)
global A opt holdon label figmain
%判断数据系列的维度
if isempty(A) errordlg ('No data present. Sure you read a file yet?',
'Error'); return; end
if size(A,2) < 3 errordlg ([ 'Seems to be only ',num2str(size(A,2)),'
columns of data. Need at least 3'],'Error'); return; end
%绘制图形
figure (figmain);
if (holdon)   hold on; else   hold off; end
%读入数据系列
X= A(:,opt.xc(1));
Y= A(:,opt.yc(1));
M= A(:,opt.zc);
%当数据系列的维度不匹配时，显示错误信息
if size(M,2) > 1 & size(M,2) < length(X)
    warn= msgbox('Number of Z columns > 1 but < number of rows in X.
Only the 1st Z column displayed','Warning');
    waitfor(warn);
    M= A(:,opt.zc(1));
end

if size(M,2) == 1
   [XI,YI]= meshgrid(min(X):range(X)/(length(X)-1):max(X),min(Y):range
(Y)/(length(Y)-1):max(Y));
    Mitp= griddata(X,Y,M,XI,YI);
end
%选择绘制图形的图表类型
switch type
    case 'waterfall'
         if size(M,2) ~= 1 waterfall(X,Y,M); end
         if size(M,2) == 1 waterfall(XI,YI,Mitp);
         end
    case 'ribbon'
         ribbon(X,M);
    case 'grid'
         if size(M,2) ~= 1 mesh(X,Y,M); end
```

```
            if size(M,2) == 1 mesh(XI,YI,Mitp);
            end
        case 'bar3'
            if size(M,2) ~= 1 bar3(X,M,'detached'); end
            if size(M,2) == 1 bar3(X,Mitp,'detached'); end
        case 'plot3'
            plot3(X,Y,M,'o');
        case 'stem3'
            stem3(X,Y,M(:,1));
        case 'surface'
            if size(M,2) ~= 1 surf(X,Y,M); end
            if size(M,2) == 1 surf(XI,YI,Mitp);
            end
        case 'smooth'
            if size(M,2) ~= 1 surfl(X,Y,M); end
            if size(M,2) == 1 surfl(XI,YI,Mitp); end
            shading interp;
            colormap(pink);
        case 'contour'
            if size(M,2) ~= 1 contour(X,Y,M,10); end
            if size(M,2) == 1 contour(XI,YI,Mitp,10); end
    end
    %添加图表的标题和坐标轴名称
    title(label.t); xlabel(label.x); ylabel(label.y); zlabel(label.z);
    grid on;
    return
```

将上面的程序代码保存为"plot3d.m"，该程序代码将是Graphs菜单选项中"3D"的菜单选项对应的程序代码。

添加主调函数

延续上面小节的步骤。

step 1 单击MATLAB命令窗口工具栏中的□按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
function Qplot()
clear all
warning off
global A cds opt holdon leg label figmain
% opt.xc= 1; opt.yc= 2; opt.ec= 3; opt.zc= 3
%创建opt和label结构体
opt= struct('xc',1,'yc',2,'ec',3,'zc',3);
label= struct('x','X','y','Y','z','Z','t','Title');
holdon= 0;
fullm= 0;
% leg= struct('tex','');
fp = get(0,'defaultfigureposition');
% fp = [ fp(1)-150 fp(2)+fp(4)-1 150 1];
%创建原始的空白图形窗口
figmain= figure ('menubar','None','Toolbar','figure','Name','Qplot',
'Resize','On','NumberTitle','off','Color','white','Position',fp);
%创建"File"菜单选项
mfile= uimenu('Label','File');
```

```
    uimenu(mfile,'Label','Read Text File','Callback','getdata(1)',
'Accelerator','R');
    uimenu(mfile,'Label','Read Excel File: All','Callback','getdata(2)
','Accelerator','E');
    uimenu(mfile,'Label','Read Excel File: Selected Data','Callback',
'getdata(3)','Accelerator','I');
    uimenu(mfile,'Label','Save as...','Callback','savefigas',
'Accelerator','S');
    uimenu(mfile,'Label','Quit','Callback','exit','Separator','on',
'Accelerator','Q');
%创建 "Options" 菜单选项
mopt= uimenu('Label','Options');
    uimenu(mopt,'Label','X column  (def. 1)','Callback','getcols(''x'')
','Accelerator','X')
    uimenu(mopt,'Label','Y columns','Callback','getcols(''y'')',
'Accelerator','Y')
    uimenu(mopt,'Label','Z column','Callback','getcols(''z'')',
'Accelerator','Z')
    uimenu(mopt,'Label','Error column','Callback','getcols(''e'')')
    uimenu(mopt,'Label','Axis Labels','Callback','getlabel',
'Accelerator','L')
    uimenu(mopt,'Label','Add graph to plot','Callback','window',
'Accelerator','A')
%创建 "Graphs" 菜单选项
graph= uimenu('Label','Graphs');
%创建 "2D" 子菜单选项
    m2d= uimenu(graph,'Label','2D');
        uimenu(m2d,'Label','XY Scatter','Callback','plot2d
(''xyscatter'')');
        uimenu(m2d,'Label','XY Line','Callback','plot2d(''xyline'')');
        uimenu(m2d,'Label','XY Line with error bar','Callback','plot2d
(''error'')');
        uimenu(m2d,'Label','Horizontal Bar (grouped)','Callback',
'plot2d(''hbarg'')');
        uimenu(m2d,'Label','Horizontal Bar (stacked)','Callback',
'plot2d(''hbars'')');
        uimenu(m2d,'Label','Vertical Bar (grouped)','Callback','plot2d
(''vbarg'')');
        uimenu(m2d,'Label','Vertical Bar (stacked)','Callback','plot2d
(''vbars'')');
        uimenu(m2d,'Label','Vertical Bar with error bars','Callback',
'plot2d(''barerror'')');
        uimenu(m2d,'Label','Histogram','Callback','plot2d(''hist'')');
        uimenu(m2d,'Label','Stem','Callback','plot2d(''stem'')');
        uimenu(m2d,'Label','Stairs','Callback','plot2d(''stairs'')');
        uimenu(m2d,'Label','Rose','Callback','plot2d(''rose'')');
        uimenu(m2d,'Label','Polar','Callback','plot2d(''polar'')');
        uimenu(m2d,'Label','Compass','Callback','plot2d(''compass'')');
        uimenu(m2d,'Label','Pie','Callback','plot2d(''pie'')');
%创建 "3D" 子菜单选项
m3d= uimenu(graph,'Label','3D');
        uimenu(m3d,'Label','Scatter 3D','Callback','plot3d(''plot3'')
');
        uimenu(m3d,'Label','Stem 3D','Callback','plot3d(''stem3'')');
        uimenu(m3d,'Label','Bar 3D','Callback','plot3d(''bar3'')');
        uimenu(m3d,'Label','Waterfall','Callback','plot3d
```

```
[illegible]
uimenu(m3d,'Label','Grid','Callback','plot3d('grid')');
uimenu(m3d,'Label','Surface','Callback','plot3d('surface')');
uimenu(m3d,'Label','Smooth Surface','Callback','plot3d
[illegible]
[illegible]
```

[illegible]

Step 2 [illegible]

```
mopt= uimenu('Label','Options');
[illegible]
uimenu(mopt,'[illegible]','[illegible]','[illegible]','[illegible]','[illegible]',
'[illegible]','[illegible]')
```

[illegible]

在 MATLAB 中，菜单选项的快捷键只在菜单选项不可见时才有效。

11.6.6 演示 GUI 对象

延续上面小节的步骤，

Step 1 [illegible]

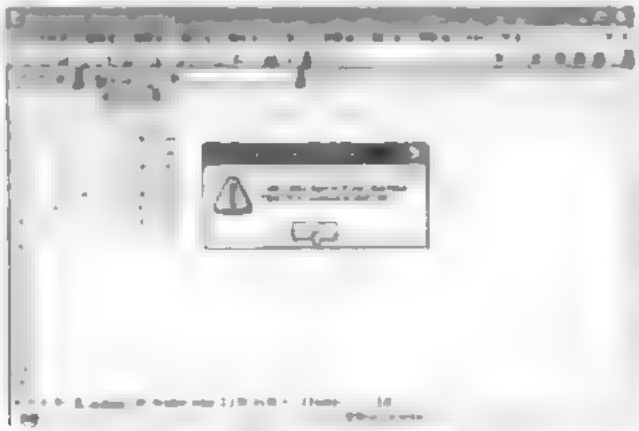


图 11.70 读入数据文件

Step 2 [illegible]

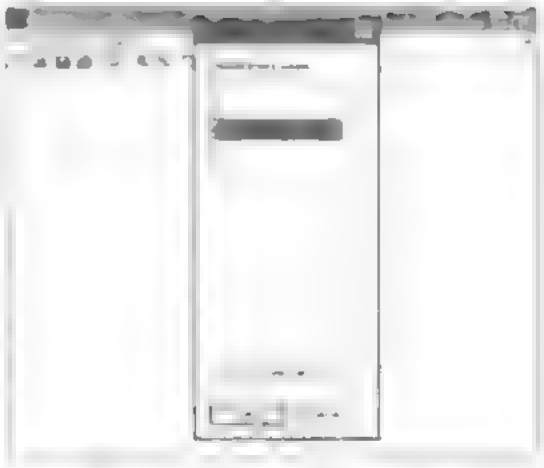


图 11.71　添加误差线数据系列

STEP 3

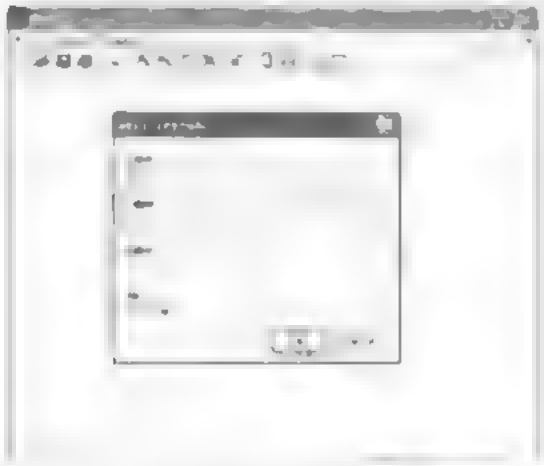


图 11.72　添加图形的标题

STEP 4

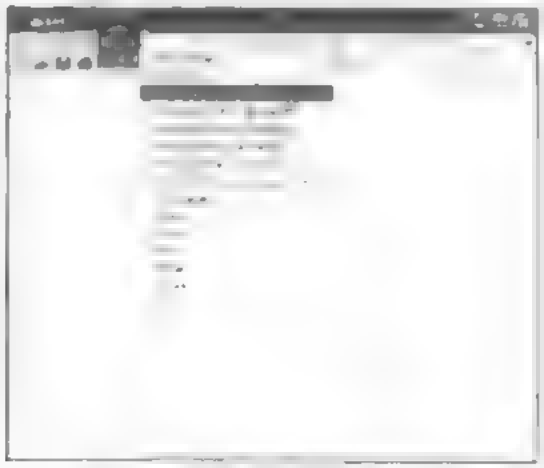


图 11.73　选择绘制的图形类型

Step 5 [illegible]

所示。

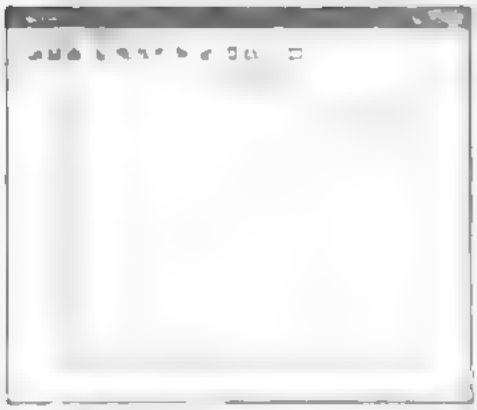


图 11.74 绘制的图形结果

Step 6 [illegible]

命令，查看重新绘制的图形结果，如图 11.75 所示。

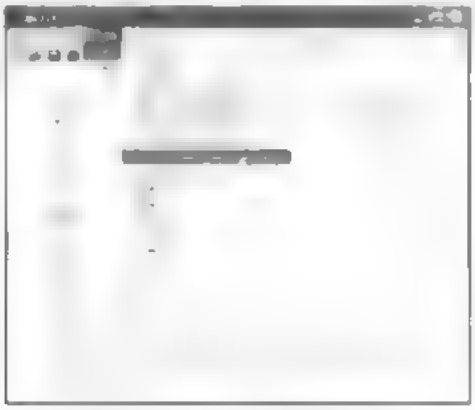


图 11.75 重新选择图形类型

Step 7 [illegible]

11.76 所示。

图 11.76 修改后的图形

[illegible]

11.7 创建现场菜单

[illegible]

创建现场菜单的一般步骤如下

- 利用命令 uicontextmenu 创建现场菜单对象
- 利用命令 uimenu 来设置该现场菜单对象的属性
- 利用命令 set 将现场菜单和图形对象联系起来。

[illegible]

11.7.1 编写 GUI 的程序代码

例 11.6 [illegible]

```
>> imagesc(peaks);
[illegible];
>> imagemenu
```

上面的程序代码就可以在图形对象中添加现场菜单，如图 11.77 所示。

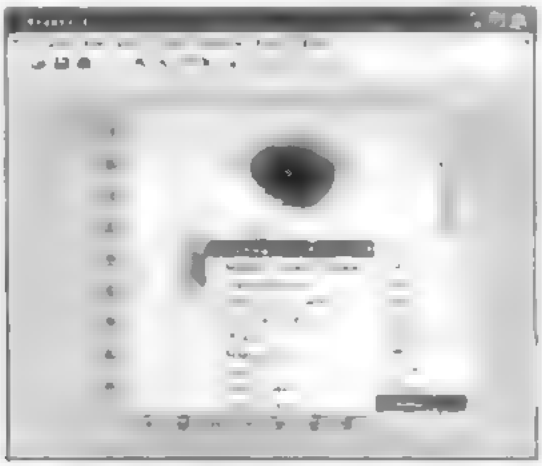


图 11.77 演示实现的现场菜单

[illegible]

step 1 单击MATLAB命令窗口工具栏中的按钮，打开M文件编辑器。在M文件编辑器中输入下面的程序代码：

```
% Menu的回调函数
function togglecolorbar(obj, eventdata)
%确定是否在图形界面中包含颜色条对象
phch = get(findall(gcf,'type','image','tag','TMW_COLORBAR'),{'parent'});
for i=1:length(phch)
    phud = get(phch{i},'userdata');
    if isfield(phud,'PlotHandle')
        if isequal(gca, phud.PlotHandle)
            delete(phch{i})
            axis image
            return
        end
    end
end

% 如果没有，则创建颜色条
colorbar
axis image
```

上面的程序代码的主要功能是在图形对象中显示颜色条，该程序代码对应的菜单选项是"Togglecolorbar"。在上面的程序代码中，首先判断图形中是否有图形颜色条，如果没有颜色条，则向其中添加颜色条对象。

step 2 在M文件编辑器中添加下面的程序代码：

```
% Menu 回调函数
function colormaplength(obj, eventdata)
%获取当前图形的色图
cmap = colormap;
%获取色图的像素单位
oldlength = length(cmap);
clength = cellstr(num2str(oldlength));
%提示用户输入新的像素单位
new = inputdlg({'Enter new colormap length:'}, ...
    'New colormap length', 1, clength);
newlength = str2double(new{1});
oldsteps = linspace(0, 1, oldlength);
newsteps = linspace(0, 1, newlength);
newmap = zeros(newlength, 3);

for i=1:3
    % 在RGB图形体系下对图像的像素进行插值
    newmap(:,i) = min(max(interp1(oldsteps, cmap(:,i), newsteps)', 0),
1);
end
%采用新的像素单位
colormap(newmap);
%如果在图形界面中包含了颜色条，对其更新
phch = get(findall(gcf,'type','image','tag','TMW_COLORBAR'),{'parent'});
for i=1:length(phch)
    phud = get(phch{i},'userdata');
```

```
    if isfield(phud,'PlotHandle')
        if isequal(gca, phud.PlotHandle)
            colorbar
        end
    end
end
```

上面代码的功能是重新设置图形像素的单位，该程序代码对应的菜单选项是“Colormap length”。当用户选择该菜单选项后，MATLAB就会显示出对话框，提示用户输入图形像素的新单位，确定新的图形像素后，将使用该单位绘制图形。

step 3 在M文件编辑器中添加下面的程序代码：

```
% Menu 回调函数
function call3d(obj, eventdata)
%返回当前图形的坐标轴对象
ax = gca;
temp = double(get(gco, 'CData'));
%创建新的图形窗口
newfig = figure;
newax = axes;
if isempty(get(get(ax, 'Parent'), 'Name'))
    set(newfig, 'Name','3D view');
else
    set(newfig, 'Name', [ get(get(ax, 'Parent'), 'Name') ', 3D view']);
end
%绘制曲面图
s = surf(temp, 'LineStyle', 'none');
hl = camlight;
%添加坐标轴名称
xlabel('X distance [pixels]');
ylabel('Y distance [pixels]');
axis('tight')
```

上面代码的功能是绘制当前图形数据的三维图形，该程序代码对应的菜单选项是“3D plot”。当用户选择该菜单选项后，MATLAB就会在新的图形窗口中绘制新图形，而且默认情况下的图形类型是surf曲面图。

step 3 在M文件编辑器中添加下面的程序代码：

```
% Menu 回调函数
function imagelimits(obj, eventdata)
%获取colormap像素数值范围
lims = get(gca, 'CLim');
oldlower = num2str(lims(1));
oldupper = num2str(lims(2));
%显示对话框，提示用户输入新的数值范围
new = inputdlg({'Enter new lower limit:', 'Enter new upper limit:'}, ...
    'New image limits', 1, {oldlower, oldupper});
if ~isnan(str2double(new{1})) & ~isnan(str2double(new{2}))
    set(gca, 'CLim', [ str2double(new{1}) str2double(new{2})]);
end

% 如果图形窗口中有颜色条，更新该颜色条
phch = get(findall(gcf,'type','image','tag','TMW_COLORBAR'),{'parent'});
```

```
for i=1:length(phch)
    phud = get(phch{i},'userdata');
    if isfield(phud,'PlotHandle')
        if isequal(gca, phud.PlotHandle)
            colorbar
        end
    end
end
```

上面代码的功能是设置图形中colormap数值范围，对应的菜单选项为"Image limits"。当用户选中该选项后，MATLAB会显示相应的对话框，用户可以在对话框中设置上限和下限，然后系统会按照新的数值范围绘制图形。

step 5 在M文件编辑器中添加下面的程序代码：

```
% Menu 回调函数
function titlecallback(obj, eventdata)
%获取原始图形窗口的标题属性
old = get(gca, 'title');
oldstring = get(old, 'string');
if ischar(oldstring)
    oldstring = cellstr(oldstring);
end
%提示用户输入新的标题
new = inputdlg('Enter new title:', 'New image title', 1, oldstring);
%设置新的标题
set(old, 'string', new);

% Menu 回调函数
function xaxiscallback(obj, eventdata)
%获取原始图形窗口的x轴名称
old = get(gca, 'xlabel');
oldstring = get(old, 'string');
if ischar(oldstring)
    oldstring = cellstr(oldstring);
end
%输入新的X轴名称
new = inputdlg('Enter new X-axis label:', 'New image X-axis label', 1,
oldstring);
%设置新的x轴名称
set(old, 'string', new);

% Menu callback
function yaxiscallback(obj, eventdata)
%获取原始图形窗口的y轴名称
old = get(gca, 'ylabel');
oldstring = get(old, 'string');
if ischar(oldstring)
    oldstring = cellstr(oldstring);
end
%输入新的y轴名称
new = inputdlg('Enter new Y-axis label:', 'New image Y-axis label', 1,
oldstring);
%设置新的y轴名称
set(old, 'string', new);
```

上面代码的功能是设置图形的标题和X、Y坐标轴的名称，对应的函数分别为titlecallback、xaxiscallback和yaxiscallback，对应的菜单选项为“Title”、“X axis Label”和“Y axis Label”。当用户选用该菜单选项后，会弹出相应的对话框，用户可以在对话框中设置图形的标题、X坐标轴和Y坐标轴的名称。

step 6 在M文件编辑器中添加下面的程序代码：

```
function imagemenu(handle)
%  示例
%   imagesc(peaks)
%   axis image
%   imagemenu
if nargin == 0
    % Use all images in current figure as default
    handle = gcf;
end

handle = findobj(handle, 'type', 'image');

% 定义现场菜单
cmenu = uicontextmenu;
% 定义现场菜单子选项
colormapmenu = uimenu(cmenu, 'Label', 'Colormap');
uimenu(cmenu, 'Label', 'Reverse current colormap', 'Callback', 'colormap
(flipud(colormap))');
uimenu(cmenu, 'Label', 'Toggle colorbar', 'Callback', @togglecolorbar);
if exist('pixval.m')
        uimenu(cmenu, 'Label', 'Toggle pixel values', 'Callback',
'pixval');
end
uimenu(cmenu, 'Label', 'Colormap length...', 'Callback', @colormaplength);
uimenu(cmenu, 'Label', '3D plot...', 'Callback', @call3d);
uimenu(cmenu, 'Label', 'Image limits...', 'Callback', @imagelimits);
uimenu(cmenu, 'Label', 'Title...', 'Callback', @titlecallback);
uimenu(cmenu, 'Label', 'X-axis label...', 'Callback', @xaxiscallback);
uimenu(cmenu, 'Label', 'Y axis label...', 'Callback', @yaxiscallback);

% 定义 "colormapmenu" 菜单的子选项
uimenu(colormapmenu, 'Label', 'Jet', 'Callback', 'colormap(jet)');
uimenu(colormapmenu, 'Label', 'Gray', 'Callback', 'colormap(gray)');
uimenu(colormapmenu, 'Label', 'Hot', 'Callback', 'colormap(hot)');
uimenu(colormapmenu, 'Label', 'Bone', 'Callback', 'colormap(bone)');
uimenu(colormapmenu, 'Label', 'Cool', 'Callback', 'colormap(cool)');
uimenu(colormapmenu, 'Label', 'Color cube', 'Callback', 'colormap
(colorcube)');
uimenu(colormapmenu, 'Label', 'HSV', 'Callback', 'colormap(hsv)');
uimenu(colormapmenu, 'Label', 'Prism', 'Callback', 'colormap(prism)');
uimenu(colormapmenu, 'Label', 'Spring', 'Callback', 'colormap(spring)');
uimenu(colormapmenu, 'Label', 'Summer', 'Callback', 'colormap(summer)');
uimenu(colormapmenu, 'Label', 'Winter', 'Callback', 'colormap(winter)');
% 将菜单对象添加到图形句柄中
set(handle, 'uicontextmenu', cmenu);
```

将全部的程序代码保存为“imagemenu.m”文件。在程序代码中，首先使用uicontextmenu命令

[illegible]菜单选项和图形句柄联系起来。

11.7.2 演示 GUI 对象

继续上面小节的步骤

step 1 演示完成的程序代码。在命令窗口中输入下面的代码

```
>> imagesc(sphere(50))
>>axis image
>>imagemenu
```

[illegible]

step 2 [illegible]

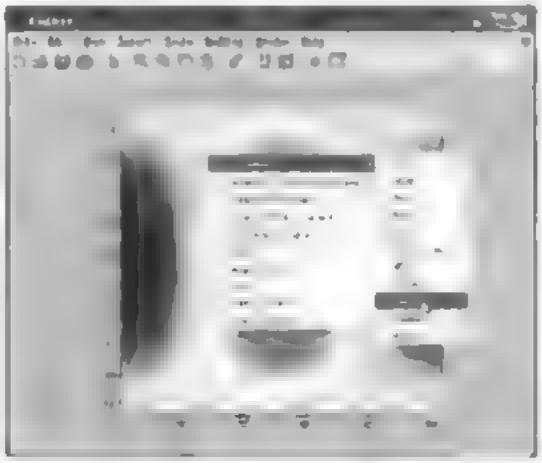


图 11.78 修改图形的颜色模式

step 3 [illegible]

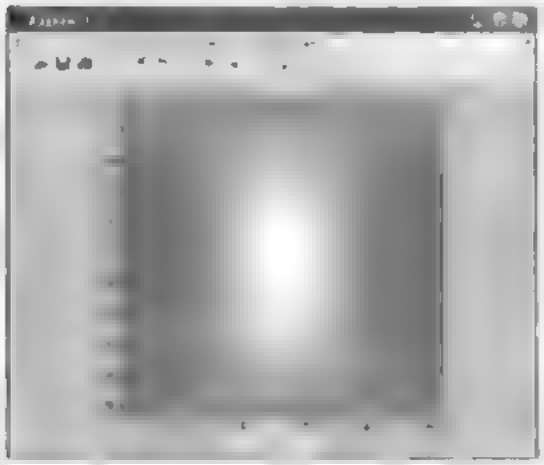


图 11.79 修改后的图形

Step 4 [illegible]

命令，如图11.80所示。

图11.80 添加颜色条

Step 5 [illegible] values' 命令，如图11.81所示。

图11.81 显示图形的像素数值

Step 6 [illegible]

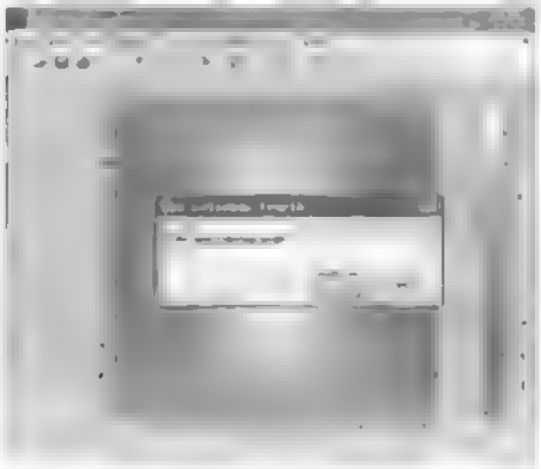


图11.82 设置好的图形像素数值

step 7 [illegible]会在新的图形界面中显示三维图形，如图 11.83 所示。

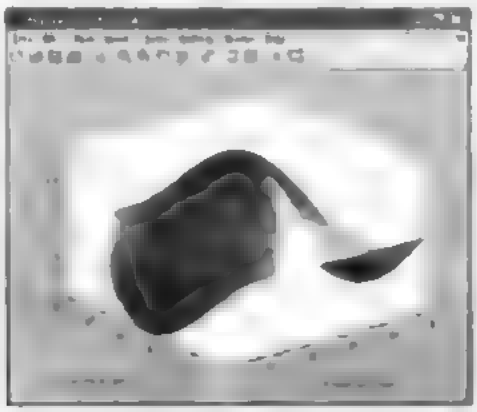


图 11.83 显示三维图形

step 8 [illegible]的上下限，如图 11.84 所示。

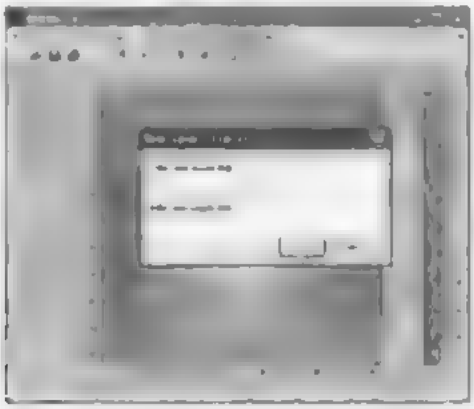


图 11.84 设置新的图形像素范围

step 9 [illegible]的名称，得到的最后结果如图 11.85 所示。

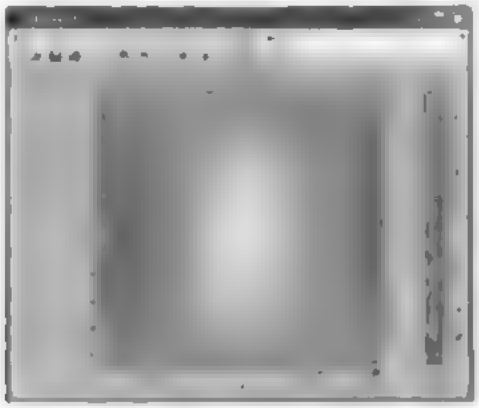


图 11.85 添加图形标题

11.8 创建GUI对象的用户控件

[illegible]

[illegible]

例11.7 编写一个关于旋转三维显示的GUI，完成后的GUI如图11.86所示。

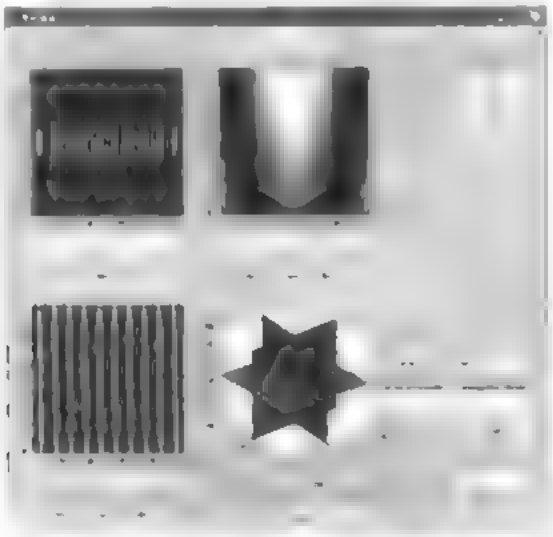


图11.86 完成的GUI对象

[illegible]

11.8.1 添加控件组件

在本小节中，将分步骤将坐标轴添加各种控件组件。

步骤1 [illegible]，将组件添加到GUIDE中，如图11.87所示。

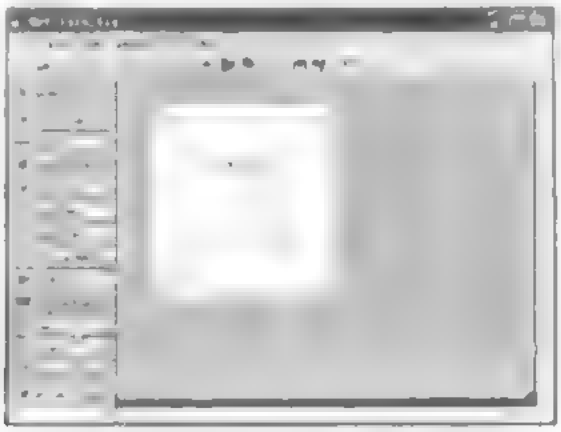


图11.87 添加坐标轴控件

step 2 [illegible]，如图 11.88 所示。

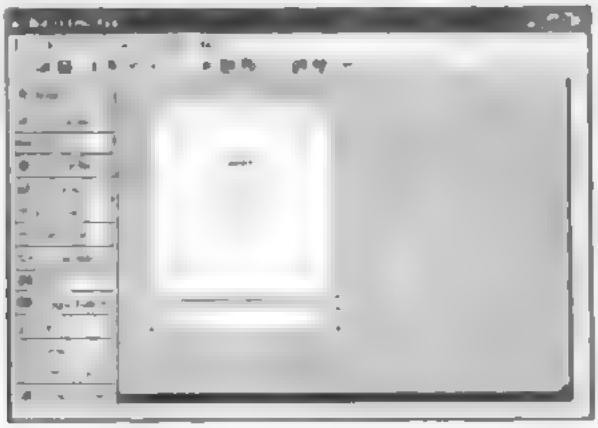


图 11.88 添加滚动条控件

[illegible]

step 3 [illegible]一个名称的属性"Value"，其对应的属性和游标位置相对应。

[illegible]

step 4 [illegible]

图 11.89 添加"静态文本"控件

[illegible]示；当用户向右移动该静态文本时，其对应的数值是减少的。

[illegible]

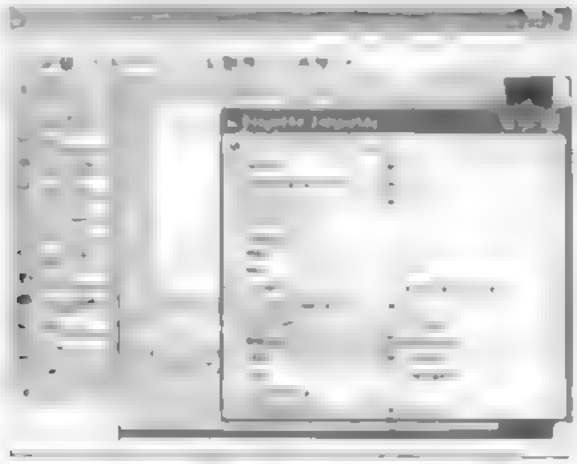


图 11.90 添加"复选框"控件

[illegible] "Transpose" 选项[illegible]

[illegible]

[illegible]

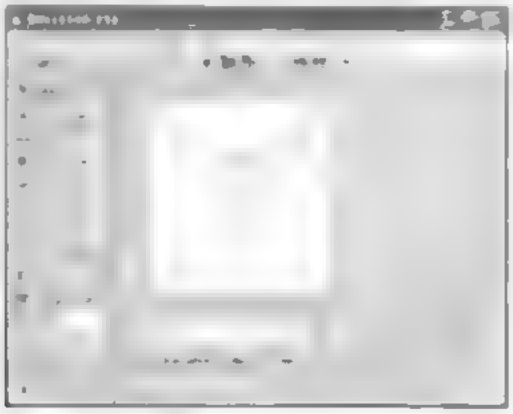


图 11.91 添加其他复选框控件

[illegible]

[illegible]

[illegible]

图 11.92 复制上面步骤中的控件

[illegible]

[illegible]

图 11.93 添加图形控件

Step 9 分析图形控件的功能。

- [illegible]
- [illegible]

[illegible]

Step 10 [illegible]

- **弹出式菜单（Pop-up Menu）**：[illegible]
- **列表框（Listbox）**：[illegible]
- **单选按钮（Radio Button）**：[illegible]

[illegible]

11.8.2 添加控件的程序代码

[illegible]

Step 1 [illegible]

图 11-94 系统控件的效果

Step 2

```
function myplot(handles,n)
vmx=max(handles.vol(:))
创建x-yz断面视角
    if get(handles.checkbox1,'value'), I=I'; end
    if get(handles.checkbox4,'value'), I=flipud(I); end
    if get(handles.checkbox7,'value'), I=fliplr(I); end
%     axes(handles.axes1);imagesc(I);
    axes(handles.axes1);
(I);end
    pbaspect('manual');pbaspect(handles.axes1,[sp2,sp3,sp1]);
end
if any(n==2)% y -- x z
    if get(handles.checkbox5,'value'), I=flipud(I); end
    if get(handles.checkbox8,'value'), I=fliplr(I); end
    axes(handles.axes2);
```

```
    if get(handles.checkbox11,'value'),imagesc(I,[vmn,vmx]);else image(
    [illegible]
    [illegible]
    [illegible]    c=get(handles.axes2,'children'); set(c(1),'cdata',I);
    [illegible]
    [illegible]
    [illegible]
if any(n==3)|| z == x y
    I=squeeze(handles.vol(:,:,s3));
    if get(handles.checkbox3,'value'), I=I'; end
    if get(handles.checkbox6,'value'), I flipud(I); end
    [illegible]
    [illegible]
    [illegible]
(I);end
    [illegible]
    pbaspect('manual');pbaspect(handles.axes4,[ sp1,sp2,sp3 ]);
    [illegible]
    set(handles.text3,'string',[ 'z=',num2str(s3)]);
    [illegible]
end;
if get(handles.checkbox10,'value'),
    [illegible]
end;
```

数据存储代码，基本要求都是对各数据的查询。

[illegible] [illegible]

回调函数名称，编辑对应的回调函数，就像 [illegible] 一样。

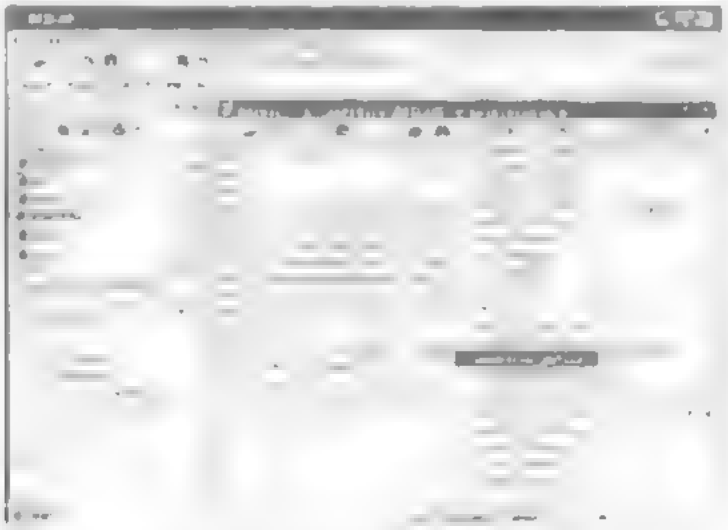


图 11.95　编写 GUI 控件的回调函数

由于篇幅限制，控件的回调函数比较复杂，下面给出部分有的回调函数的代码

```
function varargout = view3d(varargin)
```

```
%VIEW3D GUI for interactive viewing of 3D Volumes
%   VIEW3D is used view orthographic slices of 3D volumes
%   Type in an expression that generates a 3D array
%   then press the Display button
%
%   3D expressions such as: rand(50,40,30) or
%   the name or a 3D array variable in the workspace
%
%   See also: SLICE, MONTAGE, ISOSURFACE
if ~isempty(varargin) & (all(size(varargin{1})==[3 1]) | all(size(varargin
{1})==[1 3]))
    spanvar=varargin{1};
end
if nargin == 0 | exist('spanvar')% LAUNCH GUI
    fig = openfig(mfilename,'reuse');          % Generate a structure of
handles to pass to callbacks, and store it.
    handles = guihandles(fig);
    guidata(fig, handles);
    if nargout > 0
        varargout{1} = fig;
    end
        if exist('spanvar')
        set(handles.edit2,'string',num2str(spanvar(1)))
        set(handles.edit3,'string',num2str(spanvar(2)))
        set(handles.edit4,'string',num2str(spanvar(3)))
    end
elseif ischar(varargin{1}) % INVOKE NAMED SUBFUNCTION OR CALLBACK
    try
        if (nargout)
           [varargout{1:nargout}] = feval(varargin{:}); % FEVAL switchyard
        else
            feval(varargin{:}); % FEVAL switchyard
        end
    catch
        disp(lasterr);
    end
end
%| ABOUT CALLBACKS:
%| GUIDE automatically appends subfunction prototypes to this file, and
%| sets objects' callback properties to call them through the FEVAL
function varargout = slider1_Callback(h, eventdata, handles, varargin)
myplot(handles,1);
% --- Executes on slider movement.
function varargout = slider2_Callback(h, eventdata, handles, varargin)
myplot(handles,2);
% --- Executes on slider movement.
function varargout = slider3_Callback(h, eventdata, handles, varargin)
myplot(handles,3);
% --- Executes on button press in edit1.
function varargout = edit1_Callback(h, eventdata, handles, varargin)
% --- Executes on button press in pushbutton1.
function varargout = pushbutton1_Callback(h, eventdata, handles, varargin)
a=get(handles.edit1,'String');
handles.vol=double(squeeze(evalin('base',a)));
if ndims(handles.vol)~=3,
    disp('not 3d')
```

```
    return
end
[handles.sx,handles.sy,handles.sz]=size(handles.vol);
set(handles.slider1,'min',1);
set(handles.slider2,'min',1);
set(handles.slider3,'min',1);
set(handles.slider1,'max',handles.sx);
set(handles.slider2,'max',handles.sy);
set(handles.slider3,'max',handles.sz);
set(handles.slider1,'value',round(handles.sx/2)+1);
set(handles.slider2,'value',round(handles.sy/2)+1);
set(handles.slider3,'value',round(handles.sz/2)+1);
cla(handles.axes4);axis([1 handles.sx 1 handles.sy 1 handles.sz]); axis vis3d
%axes(handles.axes1);imagesc(squeeze(handles.vol(1,:,:)));axis image;
%axes(handles.axes2);imagesc(squeeze(handles.vol(:,1,:)));axis image;
%axes(handles.axes3);imagesc(squeeze(handles.vol(:,:,1)));axis image;
set(gcf,'DoubleBuffer','on');
myplot(handles,[1 2 3])
%%% produced error in matlab 7.0
%if ~isfield(handles,'clrmnu')
%    handles.clrmnu=0;
%end
%if ~handles.clrmnu;
%    colormenu;
%    handles.clrmnu=1;
%end
guidata(h,handles);
% ---------------------------------------------------------------------
function varargout = checkbox1_Callback(h, eventdata, handles, varargin)
myplot(handles,1);
% ---------------------------------------------------------------------
function varargout = checkbox2_Callback(h, eventdata, handles, varargin)
myplot(handles,2);
% ---------------------------------------------------------------------
function varargout = checkbox3_Callback(h, eventdata, handles, varargin)
myplot(handles,3);
% ---------------------------------------------------------------------
function varargout = checkbox4_Callback(h, eventdata, handles, varargin)
myplot(handles,1);
% ---------------------------------------------------------------------
function varargout = checkbox5_Callback(h, eventdata, handles, varargin)
myplot(handles,2);
% ---------------------------------------------------------------------
function varargout = checkbox6_Callback(h, eventdata, handles, varargin)
myplot(handles,3);
% ---------------------------------------------------------------------
function varargout = checkbox7_Callback(h, eventdata, handles, varargin)
myplot(handles,1);
% ---------------------------------------------------------------------
function varargout = checkbox8_Callback(h, eventdata, handles, varargin)
myplot(handles,2);
% ---------------------------------------------------------------------
function varargout = checkbox9_Callback(h, eventdata, handles, varargin)
myplot(handles,3);
% ---------------------------------------------------------------------
function varargout = pushbutton2_Callback(h, eventdata, handles, varargin)
```

```
helpdlg({'3D Volume Orthoslice Viewer','(c) Ghassan Hamarneh 2002-2004'})
% ---------------------------------------------------------------------
function varargout = pushbutton3_Callback(h, eventdata, handles, varargin)
s1=round(get(handles.slider1,'value'));
s2=round(get(handles.slider2,'value'));
s3=round(get(handles.slider3,'value'));
axes(handles.axes4); hslc=slice(handles.vol,s1,s2,s3);%rotate3d on;
axis tight; set(hslc(1:3),'LineStyle','none');
xlabel 'x' ;ylabel 'y' ;zlabel 'z';
% ---------------------------------------------------------------------
function varargout = checkbox10_Callback(h, eventdata, handles, varargin)
myplot(handles,[1 2 3]);
% ---------------------------------------------------------------------
function varargout = pushbutton4_Callback(h, eventdata, handles, varargin)
cla(handles.axes4);
% ---------------------------------------------------------------------
function varargout = edit2_Callback(h, eventdata, handles, varargin)
myplot(handles,[1 2 3]);
% ---------------------------------------------------------------------
function varargout = edit3_Callback(h, eventdata, handles, varargin)
myplot(handles,[1 2 3]);
% ---------------------------------------------------------------------
function varargout = edit4_Callback(h, eventdata, handles, varargin)
myplot(handles,[1 2 3]);
% ---------------------------------------------------------------------
function pushbutton5_Callback(hObject, eventdata, handles)
helpdlg({'- Type in an expression that generates a 3D array                    ',...
        '   then press the Load  button                                        ',...
        '- 3D expressions such as: rand(50,40,30) or                           ',...
        '    the name or a 3D array variable in the workspace                  ',...
        '- The 3 views (all except the lower right one)                        ',...
        '    display orthographic projections                                  ',...
        '- Use the scroll bars to change the number of the slice viewed        ',...
        '- Use the transpose, flipud, or fliplr to                             ',...
        '    transpose the view, flip it vertically, or horizontally           ',...
        '- Use update 3d to view the slices in a 3D view                       ',...
        '- Check auto to obtain an automatic update of the 3D view of the slices',...
        '    (Note: this may affect performance)                               ',...
        '- Use cla to clear the 3D view                                        ',...
        '    this may improve parformance                                      ',...
        '- Change the span values to the volume''s physical dimensions         ',...
        '    so the aspect ratio is displayed properly                         ',...
        '    (note: you can use relative values                                ',...
        '    for example use 1,3,2 instead of 0.5,1.5,1.0)                     ',...
        '    then press Apply                                                  '})
function pushbutton6_Callback(hObject, eventdata, handles)
if strcmp(questdlg('Exit View3D?','View3D','Yes','No','No'),'Yes')
    close(handles.view3d);
end
function pushbutton7_Callback(hObject, eventdata, handles)
myplot(handles,[1 2 3]);
```

运行程序代码

延续上面小节的步骤。

Step 1 查看默认的程序运行结果。在 MATLAB 的命令窗口中输入 ">>view3d"，按 "Enter" 键，得出默认的程序结果，如图 11.96 所示。

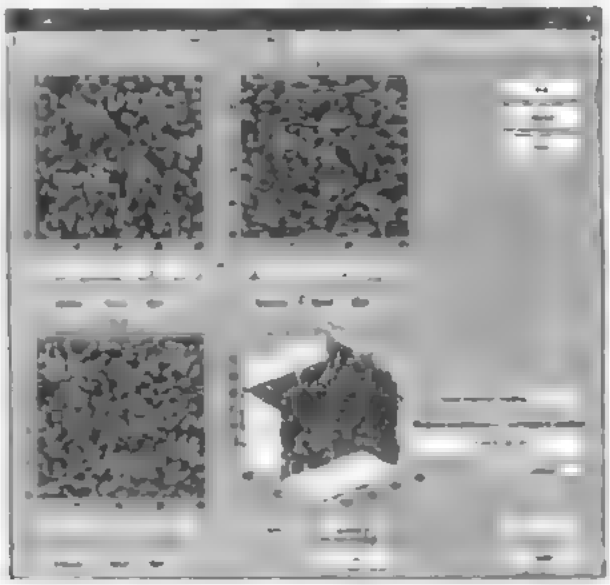


图 11.96 查看默认的程序结果

在默认情况下，绘制的三维曲面是 "ramp(-40,40)"，SLOT 选择各个散点的坐标数值构图形选项，得到的三维曲形在右下角的坐标轴系统中显示。

Step 2 在 M 文件编辑器中，输入下面的程序代码：

```
>> x=linspace(-3*pi,3*pi,1000);
>>y=x;
>> [X,Y]=meshgrid(x,y);
>>R=sqrt(X.^2+Y.^2+0.1*eps);
>>Z=sin(R)./R;
>>A=[zeros(1,1000);Z;zeros(1,1)];
>>view3d
```

Step 3 查看程序结果，输入代码后，按 "Enter" 键，得到的结果如图 11.97 所示。

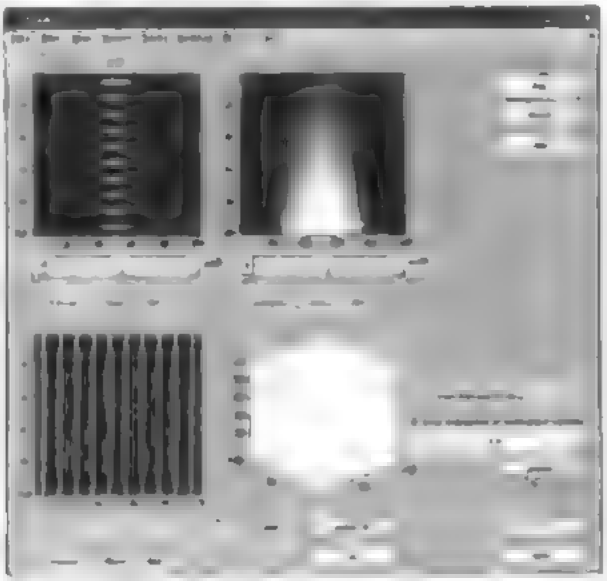


图 11.97 显示三个截面的图形

在工作空间中，用户定义了一个三维数组A3D，然后在图形界面中单击“Display”按钮，就会显示对应的截面图。

Step 4 显示三维图形，单击主窗图形界面中的“Update 3D”按钮，得到的绘图结果如图11.98所示。

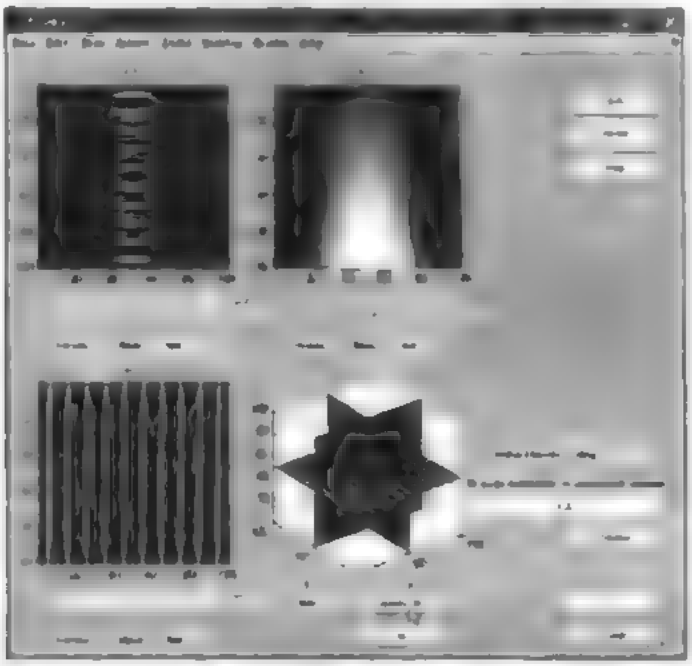


图11.98 显示三维图形

Step 5 修改截面属性，重新查看绘图结果。对于y-z截面，将其坐标数值设置为“y = 70”，同时选中“flipud”选项；对于x-z截面，将其坐标数值设置为“y = 50”，同时选中“transpose”选项；对于x-y截面，将其坐标数值设置为“z = 40”，同时选中“fliplr”选项。设置这些属性选项后，单击“Update 3D”选项，得到的结果如图11.99所示。

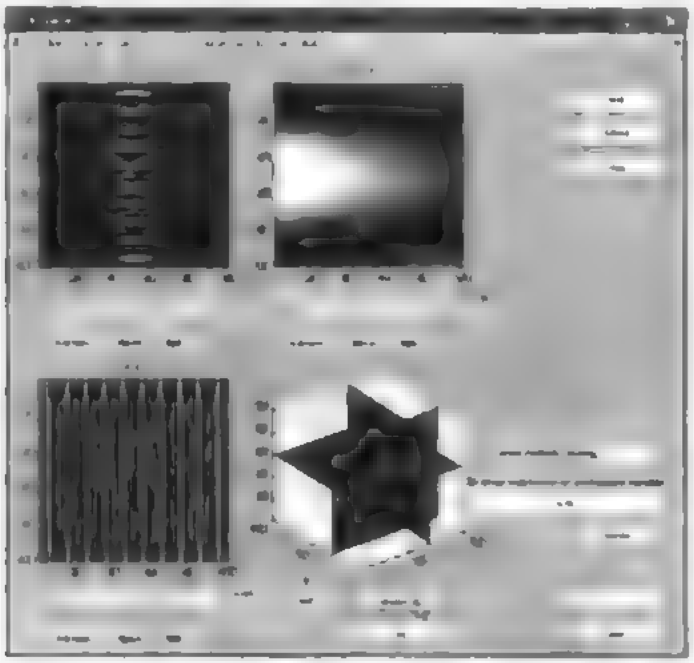


图11.99 修改后的图形界面

Step 6 修改显示比例。在文本框中输入缩放窗口显示比例，得到不同的绘图效果，[illegible]。

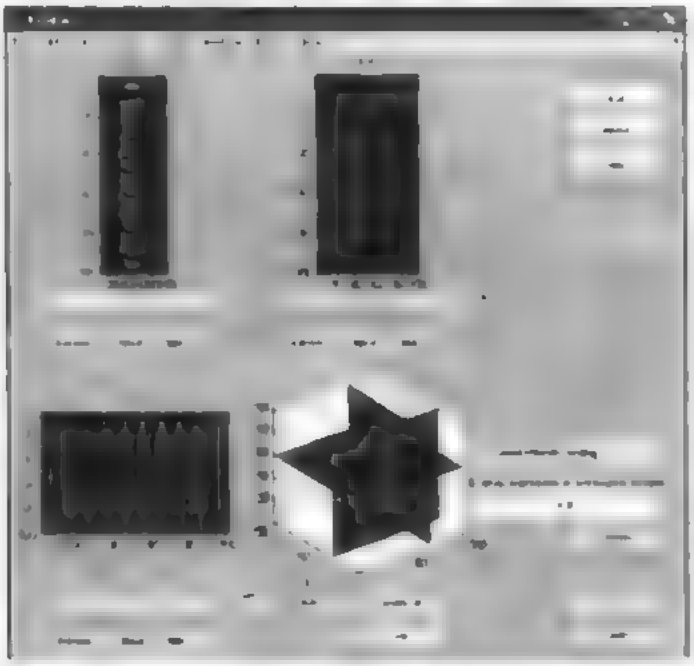


图 11.100　修改图形窗口的显示比例

[illegible]

step 7 查看三维图形。为了让读者更加直观地了解三维图形的结构，[illegible]在 MATLAB 的命令窗口中输入如下代码

```
>> x=linspace(-3*pi,3*pi,100);
>>y=x;
>> [X,Y]=meshgrid(x,y);
>>R=sqrt(X.^2+Y.^2)+eps;
>>Z=sin(R)./R;
>>surf(X,Y,Z)
>>shading interp ;colormap hsv
>>colorbar
```

step 8 查看绘制结果。输入程序代码后，按"Enter"键，得到的结果如图 11.101 所示。

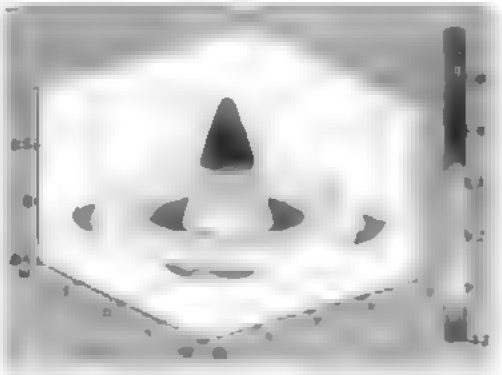


图 11.101　函数的曲面图

[illegible]

11.9 综合案例

前面介绍了如何在 MATLAB 中创建 GUI 对象的方法，并介绍了如何创建 GUI 菜单、工具栏、控件的具体方法和注意事项。本节将介绍一个比较综合的案例，说明如何综合利用这些方法创建一个比较复杂的 GUI 对象。

具体来讲，该 GUI 对象主要功能是对全局变量 [illegible] 函数的[illegible]进行绘制，以第 [illegible] 章数组为例。得到的切片图如图 11.102 所示。

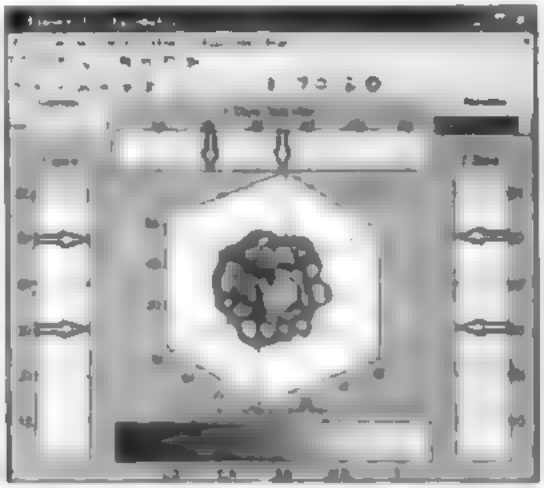


图 11.102 完成的 GUI 图形界面

11.9.1 分析 GUI 对象

[illegible]

[illegible]

下面详细分析该 GUI 中的各种对象的具体属性。

◆ 自定义菜单选项

前面[illegible]在 MATLAB 中[illegible]菜单中[illegible]去。在本例对象中，菜单条选项包括"help"菜单选项，该菜单选项会启动 MATLAB 的帮助系统。对于各选项的创建方法将在后面章节中详细介绍。

◆ 自定义工具栏

在本 GUI 对象中，工具栏的菜单选项[illegible]工具栏，而不是 MATLAB 中的默认菜单工具栏。[illegible]选择该工具栏选项，[illegible]查看该 GUI 中图形对象的常用工具选项。

◆ 自定义用户控件

在本 GUI 对象中，用户控件包括两个下拉菜单，分别用来选择图形的色阶（colormap）系统和透

明像（Alphamap）选项，这些选项都将直接影响当前对象的显示结果。

下面将详细介绍该GUI对象的操作步骤。

- 首先在MATLAB命令行中创建一个三维数组[X,Y,Z]和一个三维向量V，然后输入命令sliceomatic(V)或者sliceomatic(V,X,Y,Z)，调用该GUI对象。
- 通过鼠标来确定沿三维坐标轴方向的坐标数值，在按住X、Y坐标方向的坐标数值，该GUI允许用户在各个方向上分别切出多个切面，当选择某个切面时，MATLAB会显示对应坐标条件下的切面图形。
- 可以在各个方向，菜单中选择用户系统标准显示色彩数值，来为实现不同的颜色和透明度。
- 可以使用预期工具栏的切片按钮、缩放按钮、旋转按钮、拖动按钮等。
- 可以使用菜单选项来设置图形显示的各种属性。
- 可以使用help来查看关于该命令的帮助信息。

[illegible]

11.9.2 规划GUI的设计过程

根据前面的[illegible]，本实例整体比较复杂，由于各个对象比较繁杂，为了有效地管理整个创建对象，在创建过程之前，[illegible]，为了设置GUI中各个控件的属性，需要为每建控件编写[illegible]M文件，同时，还调用各控件系数编写主函数的程序代码。

为了[illegible]，将所有的函数的M文件保存在名为'[illegible]'文件夹中，主函数则是保存在主文件中。下面分别介绍如何创建各种控件。

11.9.3 创建GUI的工具栏对象

从本小节开始，将介绍如何在MATLAB中创建该GUI对象。

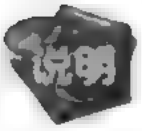
例11.8 创建11.9.1小节中的GUI对象。

Step 1 单击命令窗口工具栏的[illegible]，或者选择菜单栏的'File'→'New'→'M-file'命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码。

```
function outd = figtoolbar(d)
%创建图形的工具栏对象
%检查工具栏是否存在，如果不存在则创建
[illegible],'[illegible]','[illegible]',
if exist('uitoolfactory') ==
    %创建工具栏对象并设置工具栏属性
    d.toolbar = uitoolbar('parent',gcf);
    uitoolfactory(d.toolbar, 'Annotation.InsertRectangle');
    uitoolfactory(d.toolbar, 'Annotation.InsertEllipse');
    uitoolfactory(d.toolbar, 'Annotation.InsertTextBox');
    uitoolfactory(d.toolbar, 'Annotation.InsertArrow');
    uitoolfactory(d.toolbar, 'Annotation.InsertLine');
    uitoolfactory(d.toolbar, 'Exploration.Zoomin');
    uitoolfactory(d.toolbar, 'Exploration.Zoomout');
```

```
    uitoolfactory(d,'toolbar','Exploration.Pan'),
uitoolfactory(d,'toolbar','Exploration.Rotate');
%设置工具栏的照明属性
    cameratoolbar('show'),
    cameratoolbar('togglescenelight');
    else
    % 对R13或者更早的版本使用下面的程序代码
    try
      cameratoolbar('show');
      cameratoolbar('togglescenelight');
      %cameratoolbar('setmode','orbit');
    catch
      disp('Could not display the camera toolbar.');
    end
      end
    outd = d;
```

在上面的程序代码中，首先使用set命令将MATLAB中默认的图形窗口工具栏设置为"none"，然后使用uitoolfactory命令添加MATLAB中的默认工具栏按钮，主要有注释（annotation）和观察（exploration）两种。最后，使用cameratoolbar命令调用相关的照明工具按钮。上面程序的主要功能是为设置slicomatic对象属性选择对应的工具栏按钮。



[illegible]程序代码的最后部分，当前，MATLAB版本[illegible]程序，[illegible]用户使用的MATLAB版本[illegible]版本，[illegible]程序代码。

step 2 查看完成的图形界面。将上面的程序代码保存为"figtoolbar.m"文件，然后在命令窗口中输入"figtoolbar"，按"Enter"键，即可呈现图形如图11.103所示。

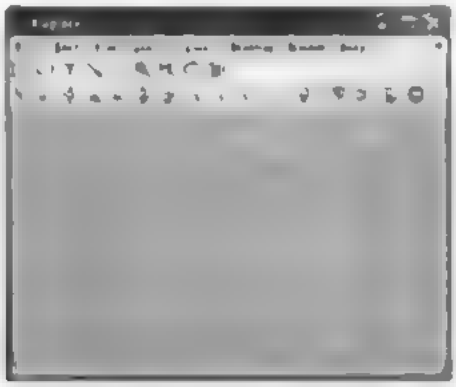


图 11.103 显示图形的工具栏对象

11.9.4 准备图形对象的基础文件

继续上面小节的步骤。

step 1 单击命令窗口"工具"中的按钮，或者选择编辑器中的"File"→"New"→"M-file"命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function appdata = slicomatic(setdata,xmesh,ymesh,zmesh)
% SLICOMATICSETDATA(rawdata) - Create the data used for
```

```
% sliceomatic in the appdata D.
% Check variables
  error(nargchk(1,4,nargin))
  % Simplify the isonormals
  disp('Smoothing for IsoNormals...');
  d.smooth=smooth3(d.data);
  d.reducenumbers=[ floor(size(d.data,2)/20)...
                    floor(size(d.data,1)/20)...
                    floor(size(d.data,3)/20) ];
  d.reducenumbers(d.reducenumbers==0)=1;
  if nargin == 4
    % Reorder vectors: make them horizontal (prepare to flipdim)
    if size(xmesh,1)>size(xmesh,2)
      xmesh=xmesh';
    end
    if size(ymesh,1)>size(ymesh,2)
      ymesh=ymesh';
    end
    if size(zmesh,1)>size(zmesh,2)
      zmesh=zmesh';
    end
    % Set axis orientation
    xdir='normal';
    ydir='normal';
    zdir='normal';
    if issorted(xmesh)~=1
      xmesh=flipdim(xmesh,2);
      xdir='reverse';
      d.xlim = [ xmesh(1) xmesh(end)];
      xmesh=flipdim(xmesh,2);
    else
      d.xlim = [ xmesh(1) xmesh(end)];
    end
    if issorted(ymesh)~=1
      ymesh=flipdim(ymesh,2);
      ydir='reverse';
      d.ylim = [ ymesh(1) ymesh(end)];
      ymesh=flipdim(ymesh,2);
    else
      d.ylim = [ ymesh(1) ymesh(end)];
    end
    % This should not be the case for medical images
    if issorted(zmesh)~=1
      zmesh=flipdim(zmesh,2);
      zdir='reverse';
      d.zlim = [ zmesh(1) zmesh(end)];
      zmesh=flipdim(zmesh,2);
    else
      d.zlim = [ zmesh(1) zmesh(end)];
    end

    % Vol vis suite takes numbers in X/Y form.
    ly = 1:d.reducenumbers(1):size(d.data,2);
    lx = 1:d.reducenumbers(2):size(d.data,1);
    lz = 1:d.reducenumbers(3):size(d.data,3);
```

```
    for i = 1:length(ly)
      ly(i) = xmesh(ly(i));
    end
    for i = 1:length(lx)
      lx(i) = ymesh(lx(i));
    end
    for i = 1:length(lz)
      lz(i) = zmesh(lz(i));
    end

    d.reducelims={ ly lx lz };
    disp('Generating reduction volume...');
    d.reduce= reducevolume(d.data,d.reducenumbers);
    d.reducesmooth=smooth3(d.reduce,'box',5);
    % Set axis
    %d.xlim = [ xmesh(1) xmesh(end)];
    %d.ylim = [ ymesh(1) ymesh(end)];
    %d.zlim = [ zmesh(1) zmesh(end)];
    d.xmesh = xmesh;
    d.ymesh = ymesh;
    d.zmesh = zmesh;
    d.xdir = xdir;
    d.ydir = ydir;
    d.zdir = zdir;

  else
    % Vol vis suite takes numbers in X/Y form.
    ly = 1:d.reducenumbers(1):size(d.data,2);
    lx = 1:d.reducenumbers(2):size(d.data,1);
    lz = 1:d.reducenumbers(3):size(d.data,3);

    d.reducelims={ ly lx lz };
    disp('Generating reduction volume...');
    d.reduce= reducevolume(d.data,d.reducenumbers);
    d.reducesmooth=smooth3(d.reduce,'box',5);

    d.xlim = [ 1 size(d.data,2)];
    d.ylim = [ 1 size(d.data,1)];
    d.zlim = [ 1 size(d.data,3)];
    d.xmesh = nan;
    d.ymesh = nan;
    d.zmesh = nan;
    d.xdir = 'normal';
    d.ydir = 'normal';
    d.zdir = 'normal';
  end

  appdata = d;
```

上面的程序代码比较复杂，下面详细介绍其具体的含义。

- 程序代码error(nargchk(1,4,nargin))……d.reducenumbers(d.reducenumbers==0)=1 的功能是判断函数的参数个数，然后将 isonormal 的数据进行归一处理。
- 程序代码 if nargin == 4……zmesh=zmesh', end 的功能是将函数的参数进行转置，将列向量转换成为行向量，这个步骤的主要目的在于为后面图形转换作准备。

- 程序代码 xdir='normal'……d.zlim = [zmesh(1) zmesh(end)]，end 的功能是，首先设置三向坐标系统的方向为正常方向，然后使用三个循环结构将数据按照图形对象结构进行转换，并转换坐标轴方向，设置坐标轴的刻度范围。
- 程序代码 ly = 1:d.reducenumbers(1):size(d.data,2) ……d.zdir = zdir 的功能是，将三维数据转换成为 X/Y 坐标轴中的数据系列，然后重新设置坐标轴的方向。
- 程序代码 else……appdata = d 的功能和上面的程序代码段类似，只是处理的情况是函数的参数个数不是 4 个的情况，原理相同，这里就不重复介绍了。

将上面的程序代码保存为“sliceomaticsetdata.m”文件，该文件将是在后面的步骤中绘制三维图形的基础文件。

step 2 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的“File” ⇨ “New” ⇨ “M-file”命令，打开一个空白的 M 文件编辑器，然后在 M 文件编辑器中输入下面的代码：

```
function sliceomaticmotion(fig,action)
% Handle generic motion events for the figure window.
  obj = hittest(fig);

  % 当用户使用鼠标在图形窗口中移动时显示光标
  if ~isempty(obj)
    t = getappdata(obj,'motionpointer');
    cc = get(fig,'pointer');
      if t
      newc = t;
    else
      newc = get(0,'defaultfigurepointer');
    end
      if isa(newc,'char') && isa(cc,'char') && ~strcmp(newc,cc)
      setpointer(fig, newc);
    end
  end
    d = getappdata(fig,'sliceomatic');
  %创建切面直线
  if isempty(d.motionmetaslice)
    d.motionmetaslice = line('parent',d.axmain,...
                             'vis','off',...
                             'linestyle','--',...
                             'marker','none',...
                             'linewidth',2,...
                             'erasemode','xor','clipping','off');
    setappdata(fig,'sliceomatic',d);
  end
  showarrowtip(obj);
    if isempty(obj) || (obj ~= d.axx && obj ~= d.axy && obj ~= d.axz)
    set(d.motionmetaslice,'visible','off');
       return
  end

  aa = obj;
  apos=get(aa,'currentpoint');
  xl = d.xlim;
  yl = d.ylim;
  zl = d.zlim;
```

```
% 获取鼠标移动处的数值坐标
    if aa==d.axx || aa==d.axiso
    if aa==d.axiso
          else
      xdata = [ apos(1,1) apos(1,1) apos(1,1) apos(1,1) apos(1,1) ];
      ydata = [ yl(1) yl(2) yl(2) yl(1) yl(1) ];
      zdata = [ zl(2) zl(2) zl(1) zl(1) zl(2) ];
    end
  else
    % 用户绘制Y向或者Z向的切片图的坐标
    if aa==d.axy
      ydata = [ apos(1,2) apos(1,2) apos(1,2) apos(1,2) apos(1,2) ];
      xdata = [ xl(1) xl(2) xl(2) xl(1) xl(1) ];
      zdata = [ zl(2) zl(2) zl(1) zl(1) zl(2) ];
    else
      zdata = [ apos(1,2) apos(1,2) apos(1,2) apos(1,2) apos(1,2) ];
      ydata = [ yl(1) yl(2) yl(2) yl(1) yl(1) ];
      xdata = [ xl(1) xl(2) xl(2) xl(1) xl(1) ];
    end
  end
  set(d.motionmetaslice,'visible','on',...
                    'xdata',xdata,'ydata',ydata,'zdata',zdata);
```

上面的程序代码的主要功能是处理鼠标在图形对象中移动时的问题。根据最后的Gui结果要求，当鼠标在各个坐标轴Slice控制面板中移动时，应该显示出对应的坐标符号，同时在显示窗口中显示对应的截面。将上面的程序代码保存为"slicecmatomotion.m"文件，该文件将是在后面的步骤中绘制三维图形的基础文件。

由于篇幅所限，本章未给出所有代码，[illegible]，请读者自行编写，或为了 setcursor、showcrosswin 等函数，[illegible]。

Step 3 单击命令窗口工具栏中的 D 按钮，或者选择编辑器中的"File"→"New"→"M-file"命令，打开一个空白的 M 文件编辑器，然后在 M 文件编辑器中输入下面的代码。

```
function activelabel(label, string)
% ACTIVELABEL(LABEL, STRING) - Create a label on GCA which is
%     active.  LABEL is the property of GCA whose label you are
%     setting.  STRING is the initial text string for the label.
  l = get(gca,label);
    set(l,'string',string);
  set(l,'buttondownfcn',@activelabelbuttondown);
function activelabelbuttondown(obj, action)
% Callback when one of our active labels is clicked on.
    set(obj,'edit','on');
```

上面的程序代码功能在于设置图形各对象标签的属性，主要在于设置标签的名称和启动标签编辑的功能。将上面的程序代码保存为"activelabel.m"，该M文件将在上面的程序代码中反复被调用。

上面的程序代码比较复杂，但是和前面的程序代码有很大的区别，前面的程序中的变量是预设定的，而上面程序代码的功能是调用函数

Step 6 单击命令窗口工具栏中的按钮，或者选择编辑栏中的“File”→“New”→“M-file”命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function slicecomtools(fig,onoff,xmesh,ymesh,zmesh,xdir,ydir,zdir)
% 检查变量属性
  error(nargchk(7,8,nargin))
% 取当前图形窗口的应用程序数据
  d = getappdata(fig, 'sliceomatic');
    if onoff
        if nargin == 8
            % 如果用户没有指定mesh对象，创建该对象
      xmesh(1) = 1;
      xmesh(2) = size(d.data,2);
      ymesh(1) = 1;
      ymesh(2) = size(d.data,1);
      zmesh(1) = 1;
      zmesh(2) = size(d.data,3);
% 设置图形坐标轴的方向
      xdir = 'normal';
      ydir = 'normal';
      zdir = 'normal';
end
% 创建图形窗口，并添加对应的图形数据
    set(0,'currentfigure',fig);
set([ d.axx d.axy d.axz] ,'handlevisibility','on');
% 设置图形窗口的坐标轴属性
        set(fig,'currentaxes',d.axx);
    set(d.axx, 'xlim',[ xmesh(1) xmesh(end)],...
               'ylim',[ 1 5]);
% 设置图形窗口中 "Slice" 控制器对象的属性
    set(d.pxx, 'vertices',[ xmesh(1) xmesh(1) -1; xmesh(end) xmesh(1)
-1; xmesh(end) 5 -1; xmesh(1) 5 -1],...
               'faces',[ 1 2 3 ; 1 3 4]);
        activelabel('title', 'X Slice Controller');
        set(fig,'currentaxes',d.axy);
% 设置xy截面切片图的控件属性
    set(d.axy, 'xlim',[ 1 5],...
               'ylim',[ ymesh(1) ymesh(end)]);
    set(d.pxy, 'vertices',[ ymesh(1) ymesh(1) -1; ymesh(1) ymesh(end)
-1; 5 ymesh(end) -1; 5 ymesh(1) -1],...
               'faces',[ 1 2 3 ; 1 3 4]);
activelabel('title', 'Y Slice');
% 设置xz截面切片图的控件属性
    set(fig,'currentaxes',d.axz);
    set(d.axz, 'xlim',[ 1 5],...
               'ylim',[ zmesh(1) zmesh(end)]);
    set(d.pxz, 'vertices',[ zmesh(1) zmesh(1) -1; zmesh(1) zmesh(end)
-1; 5 zmesh(end) -1; 5 zmesh(1) -1],...
               'faces',[ 1 2 3 ; 1 3 4]);
    activelabel('title', 'Z Slice');
    set([ d.axx d.axy d.axz] ,'handlevisibility','off');
```

```
    set(d.axx,'xdir',xdir);
    set(d.axy,'ydir',ydir);
    set(d.axz,'zdir',zdir);
  else

      end
```

上面的程序代码功能是创建图形界面的Slice控件。该GUI对象中包含三个Slice控件对象，用来选择三维图形对象的三个切片面的坐标数值。上面的程序代码比较简单，只是涉及一些图形句柄的语句，这里就不重复介绍了。

将上面的程序代码保存为“slicecontrols.m”，该程序代码的主要功能就是创建图形界面中的Slice控件对象。

step 5 单击命令窗口工具栏中的D按钮，或者选择编辑栏中的“File” ⇨ “New” ⇨ “M-file”命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function isocontrols(fig, onoff)
% 检测输入变量的个数
  error(nargchk(2,2,nargin))
  d = getappdata(fig, 'sliceomatic');
    if onoff
    lim=[ min(min(min(d.data))) max(max(max(d.data)))];
     % 设置 iso 控件的属性
    set(d.axiso,'handlevisibility','on');
    set(fig,'currentaxes',d.axiso);
    set(d.axiso, 'xlim',lim,...
                 'ylim',[1 5],...
                 'clim',lim);
% 创建图形对象
    image('parent',d.axiso,'cdata',1:64,'cdatamapping','direct',...
          'xdata',lim,'ydata',[0 5],...
          'alphadata',.6, ...
          'hittest','off');
    activelabel('title','Iso Surface Controller');
    set(d.axiso,'handlevisibility','off');
      else
    % 禁止 iso 控件的功能
     delete(findobj(d.axis,'type','image'));
  end
```

上面的程序代码功能是在图形界面的底部创建一个关于ISO的控件。当用户选择该控件中的某个数值的时候，可以设置图形界面中的ISO的属性值。将上面的程序代码保存为“isocontrols.m”文件，程序代码将在后面步骤中被调用。

step 6 单击命令窗口工具栏中的D按钮，或者选择编辑栏中的“File” ⇨ “New” ⇨ “M-file”命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function appdata=sliceomaticfigure(d,xmesh,ymesh,zmesh)
% 检测输入变量的个数
error(nargchk(1,4,nargin))
% Init sliceomatic
  figure('name','Slicematic','toolbar','none');
  lim=[ min(min(min(d.data))) max(max(max(d.data)))];
```

```
if nargin==4
    % 向量转换
    if size(xmesh,1)>size(xmesh,2)
        xmesh=xmesh';
    end
    if size(ymesh,1)>size(ymesh,2)
        ymesh=ymesh';
    end
    if size(zmesh,1)>size(zmesh,2)
        zmesh=zmesh';
    end
    % 设置坐标轴的方向
    xdir='normal';
    ydir='normal';
    zdir='normal';
    if issorted(xmesh)~=1
        xmesh=flipdim(xmesh,2);
        xdir='reverse';
    end
    if issorted(ymesh)~=1
        ymesh=flipdim(ymesh,2);
        ydir='reverse';
    end
          if issorted(zmesh)~=1
        zmesh=flipdim(zmesh,2);
        zdir='reverse';
    end
    % 更新图形的数据
    d.axmain = axes('units','normal','pos',[.2  .2 .6 .6],'box','on',...
        'ylim',[ymesh(1) ymesh(end)],...
        'xlim',[xmesh(1) xmesh(end)],...
        'zlim',[zmesh(1) zmesh(end)],...
        'clim',lim,...
        'alim',lim);
    % 设置坐标轴的方向
    set(gca,'XDir',xdir,'YDir',ydir,'ZDir',zdir);
  else
    d.axmain = axes('units','normal','pos',[.2  .2 .6 .6],'box','on',...
        'ylim',[1 size(d.data,1)],...
        'xlim',[1 size(d.data,2)],...
        'zlim',[1 size(d.data,3)],...
        'clim',lim,...
        'alim',lim);
end
%设置图形的坐标轴标签
activelabel('xlabel', 'X');
activelabel('ylabel', 'Y');
activelabel('zlabel', 'Z');
%activelabel('title', 'Data');
%设置图形的视角
daspect([1 1 1]);
view(3);
axis tight vis3d;
hold on;
grid on;
  % 依次创建四个图形控件
```

```
d.axx    = axes('units','normal','pos',[.2  .81 .6 .1],'box','on',...
                'ytick',[],'xgrid','on','xaxislocation','top',...
                'zlim',[-2 1],...
                'layer','top',...
                'color','none');
d.pxx    = patch('facecolor',[1 1 1],...
                 'facealpha',.6,...
                 'edgecolor','none',...
                 'hittest','off');
setappdata(d.axx,'motionpointer','SOM bottom');
d.axy    = axes('units','normal','pos',[.05 .05 .1 .75],'box','on',...
                'xtick',[],'ygrid','on',...
                'zlim',[-2 1],...
                'layer','top',...
                'color','none');
d.pxy    = patch('facecolor',[1 1 1],...
                 'facealpha',.6,...
                 'edgecolor','none',...
                 'hittest','off');
setappdata(d.axy,'motionpointer','SOM right');
d.axz    = axes('units','normal','pos',[.85 .05 .1 .75],'box','on',...
                'xtick',[],'ygrid','on','yaxislocation','right',...
                'zlim',[-2 1],...
                'layer','top',...
                'color','none');
d.pxz    = patch('facecolor',[1 1 1],...
                 'facealpha',.6,...
                 'edgecolor','none',...
                 'hittest','off');
setappdata(d.axz,'motionpointer','SOM left');
d.axiso  = axes('units','normal','pos',[.2 .05 .6 .1],'box','on',...
                'ytick',[],'xgrid','off','ygrid','off',...
                'xaxislocation','bottom',...
                'zlim',[-1 1],...
                'color','none',...
                'layer','top');
setappdata(d.axiso,'motionpointer','SOM top');
set([d.axx d.axy d.axz d.axiso],'handlevisibility','off');
setappdata(gcf,'sliceomatic',d);
% 创建默认的 sliceomatic控件
if nargin == 4
    slicecontrols(gcf,1,xmesh,ymesh,zmesh,xdir,ydir,zdir);
else
    slicecontrols(gcf,1);
end
    isocontrols(gcf,1);
%设置各个控件的回调函数
set(d.axx,'buttondownfcn','sliceomatic Xnew');
set(d.axy,'buttondownfcn','sliceomatic Ynew');
set(d.axz,'buttondownfcn','sliceomatic Znew');
set(d.axiso,'buttondownfcn','sliceomatic ISO');
% 设置鼠标移动的回调函数
d.motionmetaslice = [];
set(gcf,'windowbuttonmotionfcn',@sliceomaticmotion);
  % 创建工具栏
d=figtoolbar(d);
```

```
  d = figmenus(d);

  % 对图形窗口进行颜色和透明设置
  uicontrol('style','text','string','ColorMap',...
            'units','normal','pos',[0 .9 .19 .1]);
  uicontrol('style','popup','string',...
            {'jet','hsv','cool','hot','pink','bone','copper','flag',
'prism','rand','custom'},...
            'callback','sliceomatic colormap',...
            'units','normal','pos',[0 .85 .19 .1]);
  uicontrol('style','text','string','AlphaMap',...
            'units','normal','pos',[.81 .9 .19 .1]);
  uicontrol('style','popup','string',{'rampup','rampdown','vup','vdown','rand'},...
            'callback','sliceomatic alphamap',...
            'units','normal','pos',[.81 .85 .19 .1]);
% 设置文本属性
  d.tip = text('visible','off','fontname','helvetica','fontsize',10,
'color','black');
  try
      set(d.tip,'backgroundcolor',[1 1 .8],'edgecolor',[.5 .5 .5],
'margin',5);
  end
appdata = d;
```

上面的程序代码的主要功能是，依次创建各种图形对象，包括Slice控件、ISO控件、下拉菜单选项等，该程序代码将是绘制Slice对象的最主要的程序内容。最后，将上面的程序代码保存为"setvolumerange.m"文件。

处理指针对象

延续上面小节的步骤。

step 1 单击命令窗口工具栏中的D按钮，或者选择编辑栏中的"File" ⇨ "New" ⇨ "M-file"命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function setpointer(fig, ptr)
% Set the pointer on the current figure to PTR
% has several specialized SOM (SliceOMatic) pointers
  switch ptr
   case 'SOM left'
    pd = [ nan nan nan nan 1   nan nan nan nan nan nan nan nan nan nan nan
           nan nan nan 1   1   nan nan nan nan nan nan nan nan nan nan nan
           nan nan nan 1   1   nan nan nan nan nan nan nan nan nan nan nan
           nan nan 1   2   1   nan nan nan nan nan nan nan nan nan nan nan
           nan nan 1   2   1   1   1   1   1   1   1   1   1   1   1   1
           nan 1   2   2   2   2   2   2   2   2   2   2   2   2   2   1
           nan 1   2   2   2   2   2   2   2   2   2   2   2   2   2   1
           1   2   2   2   2   2   2   2   2   2   2   2   2   2   2   1
           1   2   2   2   2   2   2   2   2   2   2   2   2   2   2   1
           nan 1   2   2   2   2   2   2   2   2   2   2   2   2   2   1
           nan 1   2   2   2   2   2   2   2   2   2   2   2   2   2   1
           nan nan 1   2   1   1   1   1   1   1   1   1   1   1   1   1
```

```
            nan nan 1   2   1   nan nan nan nan nan nan nan nan nan nan nan
            nan nan nan 1   1   nan nan nan nan nan nan nan nan nan nan nan
            nan nan nan 1   1   nan nan nan nan nan nan nan nan nan nan nan
            nan nan nan nan 1  nan nan nan nan nan nan nan nan nan nan nan ];
   set(fig,'pointershapecdata', pd,...
           'pointershapehotspot', [  8 1 ] , ...
           'pointer','custom');
  case 'SOM right'
   pd = [  nan nan nan nan nan nan nan nan nan nan nan 1  nan nan nan nan
           nan nan nan nan nan nan nan nan nan nan nan 1  1   nan nan nan
           nan nan nan nan nan nan nan nan nan nan nan 1  1   nan nan nan
           nan nan nan nan nan nan nan nan nan nan nan 1  2   1   nan nan
           1   1   1   1   1   1   1   1   1   1   1   1  2   1   nan nan
           1   2   2   2   2   2   2   2   2   2   2   2  2   2   1   nan
           1   2   2   2   2   2   2   2   2   2   2   2  2   2   1   nan
           1   2   2   2   2   2   2   2   2   2   2   2  2   2   2   1
           1   2   2   2   2   2   2   2   2   2   2   2  2   2   2   1
           1   2   2   2   2   2   2   2   2   2   2   2  2   2   1   nan
           1   2   2   2   2   2   2   2   2   2   2   2  2   2   1   nan
           1   1   1   1   1   1   1   1   1   1   1   1  2   1   nan nan
           nan nan nan nan nan nan nan nan nan nan nan 1  2   1   nan nan
           nan nan nan nan nan nan nan nan nan nan nan 1  1   nan nan nan
           nan nan nan nan nan nan nan nan nan nan nan 1  1   nan nan nan
           nan nan nan nan nan nan nan nan nan nan nan 1  nan nan nan nan ];
   set(fig,'pointershapecdata', pd,...
           'pointershapehotspot', [  8 16 ] , ...
           'pointer','custom');
  case 'SOM bottom'
   pd = [  nan nan nan nan 1   1   1   1 1 1   1   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           1   1   1   1   1   2   2   2 2 2   2   1   1   1   1   1
           nan 1   1   2   2   2   2   2 2 2   2   2   2   1   1   nan
           nan nan nan 1   1   2   2   2 2 2   2   1   1   nan nan nan
           nan nan nan nan nan 1   1   2 2 1   1   nan nan nan nan nan
           nan nan nan nan nan nan nan 1 1 nan nan nan nan nan nan nan ];
   set(fig,'pointershapecdata', pd,...
           'pointershapehotspot', [  16 8 ] , ...
           'pointer','custom');
    case 'SOM top'
   pd = [  nan nan nan nan nan nan nan 1 1 nan nan nan nan nan nan nan
           nan nan nan nan nan 1   1   2 2 1   1   nan nan nan nan nan
           nan nan nan 1   1   2   2   2 2 2   2   1   1   nan nan nan
           nan 1   1   2   2   2   2   2 2 2   2   2   2   1   1   nan
           1   1   1   1   1   2   2   2 2 2   2   1   1   1   1   1
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
           nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
```

```
        nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
        nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
        nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
        nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
        nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
        nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
        nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
        nan nan nan nan 1   1   1   1 1 1   1   1   nan nan nan nan ];
set(fig,'pointershapecdata', pd,...
        'pointershapehotspot', [ 1 8 ] , ...
        'pointer','custom');
  case 'SOM leftright'
pd = [ nan nan nan nan 1   nan nan nan nan nan nan 1  nan nan nan nan
       nan nan nan 1   1   nan nan nan nan nan nan 1  1   nan nan nan
       nan nan nan 1   1   nan nan nan nan nan nan 1  1   nan nan nan
       nan nan 1   2   1   nan nan nan nan nan nan 1  2   1   nan nan
       nan nan 1   2   1   1   1   1   1   1   1   1  2   1   nan nan
       nan 1   2   2   2   2   2   2   2   2   2   2  2   2   1   nan
       nan 1   2   2   2   2   2   2   2   2   2   2  2   2   1   nan
       1   2   2   2   2   2   2   2   2   2   2   2  2   2   2   1
       1   2   2   2   2   2   2   2   2   2   2   2  2   2   2   1
       nan 1   2   2   2   2   2   2   2   2   2   2  2   2   1   nan
       nan 1   2   2   2   2   2   2   2   2   2   2  2   2   1   nan
       nan nan 1   2   1   1   1   1   1   1   1   1  2   1   nan nan
       nan nan 1   2   1   nan nan nan nan nan nan 1  2   1   nan nan
       nan nan nan 1   1   nan nan nan nan nan nan 1  1   nan nan nan
       nan nan nan 1   1   nan nan nan nan nan nan 1  1   nan nan nan
       nan nan nan nan 1   nan nan nan nan nan nan 1  nan nan nan nan ];
set(fig,'pointershapecdata', pd,...
        'pointershapehotspot', [ 8 8 ] , ...
        'pointer','custom');
  case 'SOM topbottom'
pd = [ nan nan nan nan nan nan nan 1 1 nan nan nan nan nan nan nan
       nan nan nan nan nan 1   1   2 2 1   1   nan nan nan nan nan
       nan nan nan 1   1   2   2   2 2 2   2   1   1   nan nan nan
       nan 1   1   2   2   2   2   2 2 2   2   2   2   1   1   nan
       1   1   1   1   1   2   2   2 2 2   2   1   1   1   1   1
       nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
       nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
       nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
       nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
       nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
       nan nan nan nan 1   2   2   2 2 2   2   1   nan nan nan nan
       1   1   1   1   1   2   2   2 2 2   2   1   1   1   1   1
       nan 1   1   2   2   2   2   2 2 2   2   2   2   1   1   nan
       nan nan nan 1   1   2   2   2 2 2   2   1   1   nan nan nan
       nan nan nan nan nan 1   1   2 2 1   1   nan nan nan nan nan
       nan nan nan nan nan nan nan 1 1 nan nan nan nan nan nan nan ];
set(fig,'pointershapecdata', pd,...
        'pointershapehotspot', [ 8 8 ] , ...
        'pointer','custom');
   otherwise
% Set it to the string passed in
  set(fig,'pointer', ptr);
end
```

上面程序代码的功能是设置不同情况下光标指针的形状，通过设置"PointerShapeData"属性，程序代码设置了不同的指针形状。在MATLAB中，"PointerShapeData"属性表示定义了16×16个像素组成的光标指针形状，该矩阵的元素只能选择1、2和NAN三种数值。其中，数值1代表的是黑色，数值2代表的是白色，NAN表示的是透明颜色。最后，将上面的程序代码保存为"setpointer.m"文件。

step 2 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的"File" ⇨ "New" ⇨ "M-file"命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function showarrowtip (arrow)
% Display a tip for ARROW.
% Depends on tipdata being set on the handle to ARROW.
  d=getappdata(gcf,'sliceomatic');
    if isempty(arrow)
    tipdata = [];
  else
        ctrlarrow = getappdata(arrow,'controlarrow');
        if ~isempty(ctrlarrow)
      arrow = ctrlarrow(2);
    end
        tipdata = getappdata(arrow,'tipdata');
  end
%显示光标提示的文字信息
    if ~isempty(tipdata)
      set(d.tip,'parent',tipdata.parentaxes, ...
              'string',sprintf('Value: %1.3f',tipdata.value),...
              'units','data', ...
              'position', tipdata.position, ...
              'verticalalignment', tipdata.verticalalign,...
              'horizontalalignment', tipdata.horizontalalign);
    set(d.tip,'units','pixels');
    set(d.tip,'visible','on');
      else
        set(d.tip,'visible','off');
      end
```

上面程序代码的功能是显示光标指针的尖端部分，之所以编写上面的程序代码，是为了当用户使用鼠标选择相应的对象时，显示光标提示内容。最后，将上面的程序代码保存为"snowarrowtip.m"文件。

step 2 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的"File" ⇨ "New" ⇨ "M-file"命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function [a, s]=getarrowslice
% Return the Arrow and Slice based on the GCO
  if isempty(getappdata(gco,'controlarrow')) && ...
        isempty(getappdata(gco,'isosurface'))
    a = gco;
    s = getappdata(a,'arrowslice');
    if isempty(s)
      s=getappdata(a,'arrowiso');
    end
  else
    s = gco;
```

```
    if ~isempty(getappdata(s,'isosurface'))
      s=getappdata(s,'isosurface');
    end
    a = getappdata(s,'controlarrow');
  end
```

上面程序代码的主要功能是，返回当前图形对象的箭头对象和切面对象，之所以编写上面的程序代码，其目的在于根据当前图形中的箭头对象信息进行操作。最后，将上面的程序代码保存为"getarrowslice.m"文件。

处理对象的属性

延续上面小节的步骤。

step 1 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的"File" ⇨ "New" ⇨ "M-file"命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function popset(handle,prop)
%选取句柄对象的属性域名列表
  proplist=fieldnames(get(handle(1)));
  prop=proplist{strcmpi(prop,proplist)};
  appstr = [prop '_hgstack'];
  for k=1:prod(size(handle))
        olds = getappdata(handle(k),appstr);
    if length(olds) <= 1
      error([ 'Nothing left to pop for property ' prop '.']);
    end
    set(handle(k),prop,olds{1});
    setappdata(handle(k),appstr,olds{2:end});
  end
```

上面程序代码的主要功能是，显示某个数据组中的对象属性数值。最后，将上面的程序代码保存为"popset.m"文件。

step 2 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的"File" ⇨ "New" ⇨ "M-file"命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function pushset(handle,prop,value)
%选取句柄对象的属性域名列表
  proplist=fieldnames(get(handle(1)));
  prop=proplist{strcmpi(prop,proplist)};
  appstr = [prop '_hgstack'];
  for k=1:prod(size(handle))
    oldv = get(handle(k),prop);
olds = getappdata(handle(k),appstr);
%设置句柄对象的属性值
    set(handle(k),prop,value);
    setappdata(handle(k),appstr,{ oldv olds });
  end
```

上面程序代码的主要功能是，设置新的对象属性数值。为了方便用户在后面程序中调用该代码，将上面的程序代码保存为"pushset.m"文件。

step 1 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的"File" ⇨ "New" ⇨ "M-file"命

令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function tf=propcheck(obj, prop, value)
% 检测某对象是否包含具体的属性值
  try
    v=get(obj,prop);
  catch
    tf = 0;
    return
  end
  if isa(v,class(value))
    if isa(v,'char')
      tf=strcmp(v,value);
    else
      if v==value
        tf=1;
      else
        tf=0;
      end
    end
  else
    tf=0;
  end
```

上面程序代码的主要功能是，检查图形对象是否包含某项属性值，程序代码的结构并不复杂，建议用户自行理解。最后，将该程序代码保存为"propcheck.m"文件。

step 4 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的"File" ⇨ "New" ⇨ "M-file"命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
ffunction slowset(handle, prop, value, increment)
%设置对象的属性值
  global INCREMENT;
    if nargin == 4
    INCREMENT = increment;
    if INCREMENT==0
      INCREMENT=1;
    end
  else
    INCREMENT=10;
  end
  % 获取句柄对象的属性域名类标
    proplist=fieldnames(get(handle(1)));
  tprop={ proplist{strncmpi(prop,proplist,length(prop))} };
  prop=tprop{1};
    hp = [];
    for i = 1:length(handle)
    hp(i).handle = handle(i);
    hp(i).start = get(hp(i).handle,prop);
    hp(i).end = value;
    if isnumeric(hp(i).end) && isnumeric(hp(i).start)
      hp(i).values = VectorCalc(hp(i));
else
%设置对应的属性
      set(hp(i).handle,prop,value);
```

```
        hp(i).values = [];
      end
    end
      for inc = 1:INCREMENT
      for i = 1:length(handle)
        if ~isempty(hp(i).values)
          newval = reshape(hp(i).values(inc,:,:,:),...
                           size(hp(i).start,1),...
                           size(hp(i).start,2));
%设置对应的属性
                set(hp(i).handle,prop,newval);
        end
      end
      pause(.05)
    end
    function values =  VectorCalc(hp)
    global INCREMENT;
      s = prod(size(hp.end));
      values = ones(INCREMENT,size(hp.end,1), size(hp.end,2),size(hp.end,3));
      for c = 1:s
      newval = linspace(hp.start(c),hp.end(c),INCREMENT);
      values(:,c) = newval';
    end
      values = reshape(values, INCREMENT, size(hp.end,1), size(hp.end,2), ...
                       size(hp.end,3));
```

上面程序代码的主要功能是，设置图形对象的属性值，和MATLAB内置的set函数功能类似，一次只能为某个图形对象设置单个属性值。最后，将上面的程序代码保存为“slowset.m”文件。

step 5 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的“File” ⇨ “New” ⇨ “M-file”命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function setvolumerange
% Query for a new volume range based on the sliceomatic gui
% which should be GCF
   %d=getappdata(fig,'sliceomatic');
    p=get(fig,'position');
  np=[ p(1)+20 p(2)+30 400 200];
%创建新的图形窗口
  figure('position',np);
%设置图形窗口的控件
  uicontrol('units','norm','style','text','string','X Range',...
            'position',[ 0 .6 .3 .3]);
  uicontrol('units','norm','style','text','string','Y Range',...
            'position',[ 0 .3 .3 .3]);
  uicontrol('units','norm','style','text','string','Z Range',...
            'position',[ 0 0  .3 .3]);
```

上面程序代码的功能是，重新设置三维图形数据的数值范围。当用户运行上面的程序代码时，MATLAB会显示一个新的图形界面，提示用户输入新的图形位置参数。

编写主程序代码

延续上面小节的步骤。

step 1 单击命令窗口工具栏中的□按钮，或者选择编辑栏中的"File" ⇨ "New" ⇨ "M-file"命令，打开一个空白的M文件编辑器，然后在M文件编辑器中输入下面的代码：

```
function p=arrow(parent,dir,pos)
%根据光标不同的方向设置不同的光标形状数组
 switch dir
   case 'down'
    pts=[ 0 1; -2 3; -1 3; -1 5; 1 5; 1 3; 2 3 ];
    mp = 'SOM leftright';
   case 'up'
    pts=[ 0 5; 2 3; 1 3; 1 1; -1 1; -1 3;  2 3; ];
    mp = 'SOM leftright';
   case 'right'
    pts=[ 5 0; 3 -2; 3 -1; 1 -1; 1 1; 3 1; 3 2 ];
    mp = 'SOM topbottom';
   case 'left'
    pts=[ 1 0; 3 2; 3 1; 5 1; 5 -1; 3 -1; 3 -2 ];
    mp = 'SOM topbottom';
  end
  f=[1 2 7; 3 4 5; 3 5 6 ];
 %处理光标的外观属性
  if pos(1)
    lim=get(parent,'xlim');
    fivep=abs(lim(1)-lim(2))/15/5;
    pts(:,1)=pts(:,1)*fivep+pos(1);
  elseif pos(2)
    lim=get(parent,'ylim');
    fivep=abs(lim(1)-lim(2))/15/5;
    pts(:,2)=pts(:,2)*fivep+pos(2);
  end
  % 创建patch对象
 p(1)=patch('vertices',pts,'faces',1:size(pts,1),'facec','n','edgec','k',...
             'linewidth',2,'hittest','off',...
             'parent',parent);
 p(2)=patch('vertices',pts,'faces',f,'facec','g','facea',.5,'edgec','n',...
             'parent',parent,'tag','sliceomaticarrow');
 %向patch对象中添加数据
  setappdata(p(2),'arrowcenter',pos);
  setappdata(p(2),'arrowedge',p(1));
  setappdata(p(2),'motionpointer',mp);
```

在前面的小节中，曾经专门编写过处理箭头指针对象的代码，但是本实例中还需要对箭头进行其他的处理。在上面的程序代码中，首先根据箭头指针的移动方向来显示不同的箭头形状，然后创建块对象，将箭头对象接触过的图形界面位置数据保存到块对象中。

step 2 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
function movetipforarrow(arrow, ax, value, position, va, ha)
% 显示Slice光标的数值提示内容，显示控件的数值
  tipdata.parentaxes = ax;
  tipdata.value = value;
```

```
  tipdata.position = position;
  tipdata.verticalalign = va;
  tipdata.horizontalalign = ha;
   setappdata(arrow, 'tipdata', tipdata);
  showarrowtip(arrow);
```

上面程序代码的功能是，为某个切面箭头创建当前设置，同时显示对应切面的控件数值。

step 3 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
function localcontour(slice,oldcontour,levels)
% 在切片图上绘制等高线
% 程序代码中没有引用 CONTOURSLICE 命令，而是使用特别的切片图
  d=getappdata(gcf,'sliceomatic');
cdata = get(slice,'cdata');
%获取切片图的类型信息
  st = getappdata(slice,'slicetype');
  % 计算新的等高线数值
  if nargin < 3
    if isnan(d.zmesh)==1
      c = contourc(cdata);
else
%根据切面类型的不同，绘制不同的等高线
      switch st
       case 'X'
        c = contours(d.zmesh,d.ymesh,cdata);
       case 'Y'
        c = contours(d.zmesh,d.xmesh,cdata);
       case'Z'
        c = contours(d.xmesh,d.ymesh,cdata);
      end
    end
  else
%在绘制等高线的时候，设置levels参数的数值
    if isnan(d.zmesh)==1
      c = contourc(cdata,levels);
    else
      switch st
       case 'X'
        c = contours(d.zmesh,d.ymesh,cdata,levels);
       case 'Y'
        c = contours(d.zmesh,d.xmesh,cdata,levels);
       case 'Z'
        c = contours(d.xmesh,d.ymesh,cdata,levels);
      end
    end
  end
  newvertices = [];
  newfaces = {};
  longest = 1;
  cdata = [];
  limit = size(c,2);
  i = 1;
  while(i < limit)
    z level = c(1,i);
    npoints = c(2,i);
```

```
    nexti = i+npoints+1;
    xdata = c(1,i+1:i+npoints);
ydata = c(2,i+1:i+npoints);
%根据选择的不同截面类型，计算参数vertices的数值
    switch st
     case 'X'
      xv = get(slice,'xdata');
      lzdata = xv(1,1) + 0*xdata;
      vertices = [ lzdata.', ydata.', xdata.'];
     case 'Y'
      yv = get(slice,'ydata');
      lzdata = yv(1,1) + 0*xdata;
      vertices = [ ydata.', lzdata.', xdata.'];
     case 'Z'
      zv = get(slice,'zdata');
      lzdata = zv(1,1) + 0*xdata;
      vertices = [ xdata.', ydata.', lzdata.'];
    end
    faces = 1:length(vertices);
    faces = faces + size(newvertices,1);
    longest=max(longest,size(faces,2));
    newvertices = [  newvertices ; vertices ];
    newfaces{end+1} = faces;
    tcdata =  (z_level + 0*xdata).';
    cdata = [  cdata; tcdata ]; % need to be same size as faces
        i = nexti;
  end
 %添加nan参数，结束循环
  newvertices = [  newvertices ; nan nan nan ];
  cdata = [  cdata ; nan ];
    vertmax = size(newvertices,1);
     faces = [];
  for i = 1:size(newfaces,2)
    faces = [  faces;
               newfaces{ i}  ones(1,longest-size(newfaces{ i} ,2))*vertmax
vertmax ];
  end
%设置等高线的属性
    if isempty(oldcontour)
      oldcontour  =  patch('facecolor','none',  'edgecolor',d.
defcontourcolor,...
                        'linewidth',d.defcontourlinewidth);
    try
      set(oldcontour,'linesmoothing',d.defcontoursmooth);
    catch
    end
    setappdata(slice,'contour',oldcontour);
  end
  set(oldcontour,'vertices',newvertices,...
                 'faces',faces,...
                 'facevertexcdata',cdata);
```

上面程序代码的功能是，在绘制的切面图上添加等高线。熟悉MATLAB的读者也许想知道，为何没有使用CONTOURS_ICE的内置函数。这是因为，该程序代码处理的Slice截面并不通过使用MATLAB通用代码绘制的Slice截面。

step 4 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
function p=localisosurface(volume, data, datanormals, value, oldiso)
% 处理Isosurface对象的程序代码
%设置当前图形窗口的属性
  pushset(gcf, 'pointer','watch');
%获取当前图形窗口的sliceomatic'截面的图形信息
  d=getappdata(gcf,'sliceomatic');
%绘制Isosurface对象
  fv = isosurface(volume(:),data, value);
  clim=get(gca,'clim');
%设置图形的色图属性
  cmap=get(gcf,'colormap');
  clen=clim(2)-clim(1);
  idx=floor((value-clim(1))*length(cmap)/clen);
%设置Isosurface对象的属性
 if nargin==5
    try
      set(oldiso,fv,'facecolor',cmap(idx,:));
    catch
      set(oldiso,fv,'facecolor','none');
    end
    p=oldiso;
    cap=getappdata(p,'isosurfacecap');
    if ~isempty(cap)
      localisocaps(p,cap);
    end
  else
%绘制patch对象
    if isnan(d.xmesh)==1
       p=patch(fv,'edgecolor','none','facecolor',cmap(idx,:), 'tag',
'sliceomaticisosurface');
    else
       p=patch(fv,'edgecolor','none','facecolor',cmap(idx,:),'tag',
'sliceomaticisosurface');
    end
    % d=getappdata(gcf,'sliceomatic');
    %设置图形的光照属性
    switch d.deflight
     case 'flat'
      set(p,'facelighting','flat');
     case 'smooth'
      set(p,'facelighting','phong');
    end
    setappdata(p,'isosurfacecap',[]);
  end
  setappdata(p,'isosurfacevalue',value);
  setappdata(p,'isosurfacedata',data);
  reducepatch(p,10000);
  isonormals(volume(:),datanormals,p);
%设置图形的属性
  popset(gcf,'pointer');
```

上面程序代码的功能是处理Isosurface的各种属性，具体分析如下。

- 在MATLAB中，Isosurface表示的是在某数值处和该值个个比较物体数值相等的表面，该曲面和等高线有相类似，利用这两种图表能够表示数值相等的位置。
- 必须要确定某个三维空间内某值或某范围（threshold value）的数值范围，或者需要对某一维空间内数值进行切片时，Isosurface和等高线相比有优势，需要提醒读者注意的是，该三维数据空间必须是有序的。
- 在MATLAB中，创建Isosurface曲面的第一个命令是isosurface和patch命令，而不能Isosurface曲面法线的命令是isonormals，因此，在下面的程序代码中，首先使用isosurface命令创建Isosurface曲面，然后设置曲面的表面颜色（facecolor）、边缘颜色（edgecolor）属性，最后使用isonormals创建Isosurface曲面的法线。

为了减少第3步程序代码的运行时间，在上面的程序代码中使用了reducepatch命令减少数据的"点"的数目，表面数目减少了[illegible]。[illegible]命令的详细使用方法，请读者参考MATLAB的帮助文件。

step 5 在一个面步骤3所创建的文件编辑器中，输入下面的代码：

```
function p=localisocaps(isosurface,isocap)
%处理Isocap对象
  if nargin<2 || strcmp(get(isocap,'visible'),'off')
    d=getappdata(isocap,'sliceomatic');
    isosurfacedata=getappdata(isosurface,'isosurfacedata');
    if isnan(d.xmesh)==1
      [illegible]
    else
      [illegible] xmesh,d.ymesh,d.zmesh,d.v,getappdata(isosurface,
'isosurfacevalue'));
    end
  end
  if nargin>1
    if strcmp(get(isocap,'visible'),'off')
      set(isocap,caps);
    end
    return;
  else
%绘制isocap对象
    p=patch(caps,'edgecolor','none','facecolor','flat','lighting',
            'none','tag','sliceomaticisocap');
    setappdata(p,'isosurface',isosurface);
    setappdata(p,'isosurfacecap',true);
    d=getappdata(isocap,'sliceomatic');
    %设置切片的光照属性
    switch d.shading
      case 'faceted'
        set(p,'facecolor','flat','edgecolor','black');
      case 'flat'
        set(p,'facecolor','flat','edgecolor','none');
      case 'interp'
        set(p,'facecolor','interp','edgecolor','none');
      case 'texture'
        set(p,'facecolor','flat','edgecolor','none');
      case 'none'
```

```
        set(p,'facec','none','edgec','none');
end
%设置图形的透明属性
    switch d.data.pta
     case 'none'
      [illegible]
     case 'flat'
      set(p,'facea','flat');
     case 'interp'
      set(p,'facea','interp');
     case 'texture'
      set(p,'facea','flat');
    end
end
```

上面程序中的功能是处理Isocap的各种属性，在MATLAB中，Isocap表示的是在Isosurface曲面的空间剖切平面，这些平面被称为"Isosurface"曲面的空间范围。Isocap提供Isosurface曲面内部空间的横截面视角。

在MATLAB中，[illegible]命令，[illegible]

step 6 在[illegible]中的M文件编辑器中，输入下面的代码

```
function s=local_slice(data, X, Y, Z, oldslice)
%处理切面对象
  [illegible];
  d=get(oldslice,[illegible]),
  [illegible]);
  %如果用户选取的是x向切面
  [illegible]
  %处理z向数据
    xi=round(X);
    if isnan(d.xmesh) == 1
      [illegible]
%根据需要重新获取颜色信息数组
        [illegible]
%产生绘制图形的数据格式
        [xdata ydata zdata]=meshgrid(xi,1:ds(1),1:ds(3));
%返回切面类型信息
        st = 'X';
      else
        [illegible]
      end
    else
    %如果x坐标轴的方向是反向的
      if isequal(d.xdir,'reverse')==1
%将x向数据矩阵进行转置
        locate_xi=histc(xi,fliplr(d.xmesh,2));
        slice_number=find(locate_xi);
        [illegible]
      else
```

```
        locate_xi=histc(xi,d.xmesh);
        slice_number=find(locate_xi);
      end
      if ~isempty(slice_number) && slice_number > 0 && slice_number <= ds(2)
%根据需要重新获取颜色信息数组
        cdata=reshape(data(:,slice_number,:),ds(1),ds(3));
%产生绘制图形的数据格点
        [xdata ydata zdata]=meshgrid(X,d.ymesh,d.zmesh);
%返回切面类型信息
        st = 'X';
      else
        return
      end
    end
    %如果用户选取Y向切面
    elseif ~isempty(Y)
    %处理Y向数据
     yi=round(Y);
     if isnan(d.ymesh) == 1
      if yi > 0 && yi <= ds(1)
%根据需要重新获取颜色信息数组
        cdata=reshape(data(yi,:,:),ds(2),ds(3));
%产生绘制图形的数据格点
        [xdata ydata zdata]=meshgrid(1:ds(2),yi,1:ds(3));
%返回切面类型信息
        st = 'Y';
      else
        return
      end
    else
    %如果y坐标轴的方向是反向的
      if isequal(d.ydir,'reverse')==1
%将x向数据矩阵进行转置
        locate_yi=histc(yi,flipdim(d.ymesh,2));
        slice_number=find(locate_yi);
        slice_number=length(d.ymesh)-slice_number+1;
      else
        locate_yi=histc(yi,d.ymesh);
        slice_number=find(locate_yi);
      end
      if ~isempty(slice_number) && slice_number > 0 && slice_number <= ds(1)
%根据需要重新获取颜色信息数组
        cdata=reshape(data(slice_number,:,:),ds(2),ds(3));
%产生绘制图形的数据格点
        [xdata ydata zdata]=meshgrid(d.xmesh,Y,d.zmesh);
%返回切面类型信息
        st = 'Y';
      else
        return
      end
    end
    %如果用户选取的是Z向切面
    elseif ~isempty(Z)
    %处理Z向数据
    zi=round(Z);
    if isnan(d.zmesh) == 1
```

```
      if zi > 0 && zi <= ds(3)
%根据需要重新获取颜色信息数组
        cdata=reshape(data(:,:,zi),ds(1),ds(2));
%产生绘制切面的数据格点
        [xdata ydata zdata]=meshgrid(1:ds(2),1:ds(1),zi);
%返回切面类型信息
        st = 'Z';
      else
        return
      end
    else
    %如果z坐标轴的方向是反向的
      if isequal(d.zdir,'reverse')==1
%将z向数据矩阵进行转置
        locate_zi=histc(zi,flipdim(d.zmesh,2));
        slice_number=find(locate_zi);
        slice_number=length(d.zmesh)-slice_number+1;
      else
        locate_zi=histc(zi,d.zmesh);
        slice_number=find(locate_zi);
      end
      if ~isempty(slice_number) && slice_number > 0 && slice_number <= ds(3)
%根据需要重新获取颜色信息数组
        cdata=reshape(data(:,:,slice_number),ds(1),ds(2));
%产生绘制切面的数据格点
        [xdata ydata zdata]=meshgrid(d.xmesh,d.ymesh,Z);
%返回切面类型信息
        st = 'Z';
      else
        return
      end
    end
  else
    error('Nothing was passed into LOCALSLICE.');
  end
%减少数据数组中的独维
  cdata=squeeze(cdata);
  xdata=squeeze(xdata);
  ydata=squeeze(ydata);
  zdata=squeeze(zdata);
  if nargin == 5
    % 设置原来切片面的属性
    set(oldslice,'cdata',cdata,'alphadata',cdata, 'xdata',xdata, ...
                 'ydata',ydata, 'zdata',zdata);
    s=oldslice;
    %检测图形表面的属性
    if propcheck(s,'facec','texturemap')
      textureizeslice(s,'on');
    end
    setappdata(s,'slicetype',st);
  else
    % 绘图表面图
    news=surface('cdata',cdata,'alphadata',cdata, 'xdata',xdata, ...
                 'ydata',ydata, 'zdata',zdata);
     set(news,'alphadata',cdata,'alphadatamapping','scaled','tag',
'sliceomaticslice',...
```

```
                'facelighting','none',...
                'uicontextmenu',d.uic);
s=news;
%设置截面类型
setappdata(s,'slicetype',st);
%设置图像的照明属性
    switch d.defcolor
     case 'faceted'
      set(s,'facec','flat','edgec','k');
     case 'flat'
      set(s,'facec','flat','edgec','n');
     case 'interp'
      set(s,'facec','interp','edgec','n');
     case 'texture'
      set(s,'facec','texture','edgec','n');
end
%设置图形的透明属性
    switch d.defalpha
     case 'none'
      set(s,'facea',1);
     case 'flat'
      set(s,'facea','flat');
     case 'interp'
      set(s,'facea','interp');
     case 'texture'
      set(s,'facea','texture');
    end
  end
%获取图形中等高线的数值信息
 contour = getappdata(s,'contour')
%设置等高线的属性
  if ~isempty(contour)
    try
      levels = getappdata(s, 'contourlevels');
      if isempty(levels)~=1
        localcontour(s,contour,levels);
      else
        localcontour(s, contour);
      end
    catch
      localcontour(s, contour);
    end
  end
```

在上面的程序代码中，首先判断Slice截面的数目，然后将截面数据转换为二维数据，最后，使用surface命令来创建表面对象，并设置该表面对象的各种属性，主要设置图形对象的照明属性和透明属性，具体的程序代码就不详细解释了，请读者自行分析和运行。

step 7 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
function textureizeslice(slice,onoff)
% 实现一般的切面图和材质化的切面图之间的相互转换
    for k=1:prod(size(slice))
    d=getappdata(slice(k),'textureoptimizeations');
```

```
    switch onoff
     case 'on'
%获取切面各向的数据类型
      d.xdata=get(slice(k),'xdata');
      d.ydata-get(slice(k),'ydata');
      d.zdata=get(slice(k),'zdata');
%设置材质化图形的数据类型
      setappdata(slice(k),'textureoptimizeations',d);
%返回符合材质贴图的数据数组
      if max(size(d.xdata)==1)
        nx=[d.xdata(1) d.xdata(end)];
      else
        nx=[d.xdata(1,1)   d.xdata(1,end);
            d.xdata(end,1) d.xdata(end,end)];
      end
      if max(size(d.ydata)==1)
        ny=[d.ydata(1) d.ydata(end)];
      else
        ny=[d.ydata(1,1)   d.ydata(1,end);
            d.ydata(end,1) d.ydata(end,end)];
      end
      if max(size(d.zdata)==1)
        nz=[d.zdata(1) d.zdata(end)];
      else
        nz=[d.zdata(1,1)   d.zdata(1,end);
            d.zdata(end,1) d.zdata(end,end)];
      end
%设置图形的材质化属性
      set(slice(k),'xdata',nx, 'ydata', ny, 'zdata', nz,...
                   'facec','texturemap');
      if ischar(get(slice(k),'facea'))
        set(slice(k),'facea','texturemap');
      end
      if ischar(get(slice(k),'facec'))
        set(slice(k),'facec','texturemap');
      end
     case 'off'
      if ~isempty(d)
        set(slice(k),'xdata',d.xdata,'ydata',d.ydata,'zdata',d.zdata);
        setappdata(slice(k),'textureoptimizeations',[]);
      end
%将材质化的图形转换为一般图形
      if ischar(get(slice(k),'facea')) && strcmp(get(slice(k),'facea'),
'texturemap')
        set(slice(k),'facea','flat');
      end
      if ischar(get(slice(k),'facec')) && strcmp(get(slice(k),'facec'),
'texturemap')
        set(slice(k),'facec','flat');
      end
    end
  end
```

上面的程序代码的主要功能是将正常的切片图和质地处理过的切片图进行转换。在MATLAB中，材质贴图（Texture mapping）是将一个二维图像映射到二维图像的重要手段，其程序

代码将部件数组进行转换，[illegible]转换为面要形式。

在这种图形编辑方法中，用户可以选择复杂图形对象，例如曲线、曲面等，并设置这些对象的属性和参数。[illegible]

step 8 在上面步骤打开的M文件编辑器中，输入下面的代码

```
function ss=allSlices
  ss=findobj(gcf,'type','surface','tag','sliceomaticslice');

function ss=allIsos
  ss=findobj(gcf,'type','patch','tag','sliceomaticisosurface');

function ss=allCaps
  ss=findobj(gcf,'type','patch','tag','sliceomaticisocap');
```

上面代码中使用findobj命令来查找所有标签为sliceomaticslice、sliceomaticisosurface和sliceomaticisocap的"块"对象的句柄。

step 9 在上面步骤打开的M文件编辑器中，输入下面的代码

```
function dragprep(arrowtodrag)
  arrows=findobj(gcf,'tag','sliceomaticarrow');
%设置箭形中的箭头属性，表明颜色、透明度等
  pushset(arrows,'facecolor',[1 0 0]);
  pushset(arrows,'facealpha',.2);
  pushset(arrowtodrag,'facecolor',[0 1 0]);
  pushset(arrowtodrag,'facealpha',.7);
%遍历[illegible]
  slices=allSlices;
  for i=1:length(slices)
    fa=get(slices(i),'facealpha');
    %设置所有切面的透明和边缘颜色属性
    if isa(fa,'double') && fa>.3
      pushset(slices(i),'facealpha',.3);
      pushset(slices(i),'edgecolor','n');
    else
      pushset(slices(i),'facealpha',fa);
      pushset(slices(i),'edgecolor',get(slices(i),'edgecolor'));
    end
  end
%遍历所有的isosurface对象
  isosurfs=allIsos;
  for i=1:length(isosurfs)
    %设置所有isosurface对象的透明和边缘颜色属性
    fa=get(isosurfs(i),'facealpha');
    if isa(fa,'double') && fa>.3
      pushset(isosurfs(i),'facealpha',.3);
      pushset(isosurfs(i),'edgecolor','n');
    else
      pushset(isosurfs(i),'facealpha',fa);
      pushset(isosurfs(i),'edgecolor',get(isosurfs(i),'edgecolor'));
```

```
    end
    %设置cap对象的属性
    cap=getappdata(isosurfs(i),'isosurfacecap');
    if ~isempty(cap)
      pushset(cap,'visible','off');
    end
  end
  ss=getappdata(arrowtodrag,'arrowslice');
  if isempty(ss)
    ss=getappdata(arrowtodrag,'arrowiso');
  end
%设置用户运行箭头的属性
  popset(ss,'facealpha');
  popset(ss,'edgecolor');
  pushset(gcf,'windowbuttonupfcn','sliceomatic up');
  pushset(gcf,'windowbuttonmotionfcn','sliceomatic motion');
  % Doing this makes the tip invisible when visible is on.
  showarrowtip(arrowtodrag);
```

在上面的程序代码中，首先使用程序语句设置光标的属性，然后通过上面步骤返回的对象句柄设置图形对象的属性。上面的程序代码并不复杂，请读者自行分析。

step 10 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
function dragfinis(arrowtodrag)
  arrows=findall(gcf,'tag','sliceomaticarrow');
%设置箭头对象的属性
  popset(arrowtodrag,'facecolor');
  popset(arrowtodrag,'facealpha');
  popset(arrows,'facecolor');
  popset(arrows,'facealpha');
  ss=getappdata(arrowtodrag,'arrowslice');
  if isempty(ss)
    ss=getappdata(arrowtodrag,'arrowiso');
  end
  % These pushes are junk which will be undone when all slices or
  % isosurfs are reset below.
  pushset(ss,'facealpha',1);
  pushset(ss,'edgecolor','k');
  slices=allSlices;
%设置切面属性
  if ~isempty(slices)
    popset(slices,'facealpha');
    popset(slices,'edgecolor');
  end
  isosurfs=allIsos;
%设置isosurface对象的属性
  if ~isempty(isosurfs)
    popset(isosurfs,'facealpha');
    popset(isosurfs,'edgecolor');
  end
  d=getappdata(gcf,'sliceomatic');
    if isnan(d.xmesh)==1
    for i=1:length(isosurfs)
      cap=getappdata(isosurfs(i),'isosurfacecap');
      if ~isempty(cap)
```

```
        popset(cap,'visible');
        localisocaps(isosurfs(i),cap);
      end
      if getappdata(isosurfs(i), 'reduced')
        setappdata(isosurfs(i),'reduced',0);
        localisosurface({},d.data,d.smooth,...
                        getappdata(isosurfs(i),'isosurfacevalue'),...
                        isosurfs(i));
      end
    end
  else
for i=1:length(isosurfs)
%设置cap对象的属性
      cap=getappdata(isosurfs(i),'isosurfacecap');
      if ~isempty(cap)
        popset(cap,'visible');
        localisocaps(isosurfs(i),cap);
      end
      if getappdata(isosurfs(i), 'reduced')
        setappdata(isosurfs(i),'reduced',0);
        realvolume={d.xmesh d.ymesh d.zmesh};
        localisosurface(realvolume,d.data,d.smooth,...
                        getappdata(isosurfs(i),'isosurfacevalue'),...
                        isosurfs(i));
      end
    end
  end
%设置当前图形的调用函数
  popset(gcf,'windowbuttonupfcn');
  popset(gcf,'windowbuttonmotionfcn');
  showarrowtip([]);
    buf = get(gcf,'windowbuttonupfcn');
  if ~strcmp(buf,'')
    eval(buf);
  end
```

在上面的程序代码中，首先设置了箭头光标的属性，然后将光标分为Slice截面选取光标和ISO颜色系统选取光标来分别设置属性对象。

step 11 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
function sliceomatic(p1,p2,xmesh,ymesh,zmesh)
% SLICEOMATIC - Slice and isosurface volume exploration GUI
%
% SLICEOMATIC(DATA) - Use 3D double matrix DATA as a volume data
%
% 示例
%       x = -2:.2:2; y = -2:.25:2; z = -2:.16:2;
%       [X,Y,Z,] = meshgrid(x,y,z);
%       v = X .* exp(-X.^2 - Y.^2 - Z.^2);
%       sliceomatic(v,x,y,z)
%
  if nargin==0
%当用户没有输入任何参数时，则使用下面的数据系列创建图形
    [x,y,z] = meshgrid(-2:.2:2,  2:.25:2, -2:.16:2);
    v = x .* exp(-x.^2 - y.^2 - z.^2);
```

```
    sliceomatic(v)
    return
  end
%处理输入参数
  if isa(p1,'double')
        d.data=p1;
        if nargin>=4
        if nargin==4
        zmesh=ymesh;
        ymesh=xmesh;
        xmesh=p2;
      end
%调用sliceomaticfigure函数绘制图形
      d = sliceomaticfigure(d,xmesh,ymesh,zmesh);
%调用sliceomaticsetdata函数获取图形数据
      d = sliceomaticsetdata(d,xmesh,ymesh,zmesh);
    else
      d = sliceomaticfigure(d);
      d = sliceomaticsetdata(d);
    end
        setappdata(gcf,'sliceomatic',d);
  elseif isa(p1,'char')
    % Interpret commands
    d=getappdata(gcf,'sliceomatic');
    try
      switch p1
%当参数p1的数值为xnew时
       case 'Xnew'
        if strcmp(get(gcf,'selectiontype'),'normal')
%获取当前数据点的坐标
          pt=get(gcbo,'currentpoint');
%获取当前图形的坐标轴属性
          axis(gcbo);
          X=pt(1,1);
%创建箭头对象
          newa=arrow(gcbo,'down',[X 0]);
%设置坐标轴对象的属性
          set(gcf,'currentaxes',d.axmain);
%调用localslice函数绘制切面对象
          new=localslice(d.data, X, [], []);
%设置控件箭头和切面箭头的属性数据
          setappdata(new,'controlarrow',newa);
          setappdata(newa(2),'arrowslice',new);
%设置创建的切面对象属性
           set(new,'alphadata',get(new,'cdata'),'alphadatamapping',
'scaled');
%设置箭头的属性
          set(newa,'buttondownfcn','sliceomatic Xmove');
%设置现场菜单的属性
          set([new newa],'uicontextmenu',d.uic);
          buf = get(gcf,'windowbuttonupfcn');
          if ~strcmp(buf,'')
            eval(buf);
          end
          d.draggedarrow=newa(2);
          dragprep(newa(2));
```

```
            setpointer(gcf,'SOM leftright');
            set(d.motionmetaslice,'visible','off');
          end
% 当参数 p1 的数值是 Ynew 时
% 参考前段程序代码中处理 xnew 的方法
        case 'Ynew'
          if strcmp(get(gcf,'selectiontype'),'normal')
            pt=get(gcbo,'currentpoint');
            Y=pt(1,2);
            newa=arrow(gcbo,'right',[ 0 Y]);
            set(gcf,'currentaxes',d.axmain);
            new=localslice(d.data, [], Y, []);
            setappdata(new,'controlarrow',newa);
            setappdata(newa(2),'arrowslice',new);
            set(new,'alphadata',get(new,'cdata'),'alphadatamapping','scaled');
            set(newa,'buttondownfcn','sliceomatic Ymove');
            set([ new newa],'uicontextmenu',d.uic);
            buf = get(gcf,'windowbuttonupfcn');
            if ~strcmp(buf,'')
              eval(buf);
            end
            d.draggedarrow=newa(2);
            dragprep(newa(2));
            setpointer(gcf,'SOM topbottom');
            set(d.motionmetaslice,'visible','off');
          end
% 当参数 p1 的数值是 znew 时
% 参考前段程序代码中处理 xnew 的方法
        case 'Znew'
          if strcmp(get(gcf,'selectiontype'),'normal')
            pt=get(gcbo,'currentpoint');
            Y=pt(1,2);
            newa=arrow(gcbo,'left', [ 0 Y]);
            set(gcf,'currentaxes',d.axmain);
            new=localslice(d.data, [], [], Y);
            set(new,'alphadata',get(new,'cdata'),'alphadatamapping','scaled');
            setappdata(new,'controlarrow',newa);
            setappdata(newa(2),'arrowslice',new);
            set(newa,'buttondownfcn','sliceomatic Zmove');
            set([ new newa],'uicontextmenu',d.uic);
            buf = get(gcf,'windowbuttonupfcn');
            if ~strcmp(buf,'')
              eval(buf);
            end
            d.draggedarrow=newa(2);
            dragprep(newa(2));
            setpointer(gcf,'SOM topbottom');
            set(d.motionmetaslice,'visible','off');
          end
% 当参数 p1 的数值为 ISO 时
% 参考前段程序代码中处理 xnew 的方法
        case 'ISO'
          if strcmp(get(gcf,'selectiontype'),'normal')
            pt=get(gcbo,'currentpoint');
            V=pt(1,1);
            newa=arrow(gcbo,'up',[ V 0]);
```

```
    set(gcf,'currentaxes',d.axmain);
    new=localisosurface(d.reducelims,d.reduce,d.reducesmooth,V);
    set([newa new],'uicontextmenu',d.uiciso);
    setappdata(new,'controlarrow',newa);
    setappdata(new,'reduced',1);
    setappdata(newa(2),'arrowiso',new);
    set(newa,'buttondownfcn','sliceomatic ISOmove');
    buf = get(gcf,'windowbuttonupfcn');
    if ~strcmp(buf,'')
      eval(buf);
    end
    d.draggedarrow=newa(2);
    dragprep(newa(2));
    setpointer(gcf,'SOM leftright');
  end

case 'Xmove'
  if strcmp(get(gcf,'selectiontype'),'normal')
    [a s]=getarrowslice;
    d.draggedarrow=a;
    dragprep(a);
  end
case 'Ymove'
  if strcmp(get(gcf,'selectiontype'),'normal')
    [a s]=getarrowslice;
    d.draggedarrow=a;
    dragprep(a);
  end
case 'Zmove'
  if strcmp(get(gcf,'selectiontype'),'normal')
    [a s]=getarrowslice;
    d.draggedarrow=a;
    dragprep(a);
  end
case 'ISOmove'
  if strcmp(get(gcf,'selectiontype'),'normal')
    [a s]=getarrowslice;
    d.draggedarrow=a;
    dragprep(a);
  end
case 'up'
  if strcmp(get(gcf,'selectiontype'),'normal')
    dragfinis(d.draggedarrow);
  end
case 'motion'
  % Make sure our cursor is ok
  a=d.draggedarrow;                  % 绘制箭头
  s=getappdata(a,'arrowslice');      % The slice to 'move'
  if isempty(s)
    s=getappdata(a,'arrowiso');      % or the isosurface
  end
  aa=get(a,'parent');          % 箭头的当前坐标数值
  pos=getappdata(a,'arrowcenter');
  apos=get(aa,'currentpoint');

  % 设置坐标轴的数值属性
```

```
xlimits = get(aa,'xlim');
ylimits = get(aa,'ylim');
if apos(1,1) < xlimits(1)
  apos(1,1) = xlimits(1);
elseif apos(1,1) > xlimits(2)
  apos(1,1) = xlimits(2);
end
          if apos(1,2) < ylimits(1)
  apos(1,2) = ylimits(1);
elseif apos(1,2) > ylimits(2)
  apos(1,2) = ylimits(2);
end
  if aa==d.axx || aa==d.axiso
  % We are moving an X slice
  xdiff=apos(1,1)-pos(1,1);
  v=get(a,'vertices');
  v(:,1)=v(:,1)+xdiff;
  set([ a getappdata(a,'arrowedge')],'vertices',v);
  np=[ apos(1,1) 0 ];
  % This might be a slice, or an isosurface!
  if aa==d.axiso
   new=localisosurface(d.reducelims,d.reduce,d.reducesmooth,...
                        apos(1,1),s);
    setappdata(new,'reduced',1);
   movetipforarrow(a, aa, apos(1,1), [ apos(1,1) 6 ], 'bottom','center')
  else
    %disp([ 'apos = ' num2str(apos(1,1))])
    %disp([ 'pos  = ' num2str(pos(1,1))])
    %disp([ 'change=' num2str(round(apos(1,1))~=round(pos(1,1)))]);
    if round(apos(1,1))~=round(pos(1,1))
      localslice(d.data, apos(1,1), [], [],s);
    end
   movetipforarrow(a, aa, apos(1,1), [ apos(1,1) .5 ],'top','center')
  end
else
  % 当用户移动Y向切面或者Z向切面时
  ydiff=apos(1,2)-pos(1,2);
  v=get(a,'vertices');
  v(:,2)=v(:,2)+ydiff;
  set([ a getappdata(a,'arrowedge')],'vertices',v);
  np=[ 0 apos(1,2) ];
  if aa==d.axy
    if round(apos(1,2))~=round(pos(1,2))
      localslice(d.data, [], apos(1,2), [], s);
    end
   movetipforarrow(a, aa, apos(1,2), [ 5.5 apos(1,2) ], 'middle','left');
  else
    if round(apos(1,2))~=round(pos(1,2))
      localslice(d.data, [], [], apos(1,2), s);
    end
   movetipforarrow(a, aa, apos(1,2), [ .5 apos(1,2) ], 'middle','right');
  end
end
setappdata(a,'arrowcenter',np);
try
  if isempty(get(gcf,'javaframe'))
```

```
            drawnow;
          end
        catch
          drawnow;
        end
      % IsoSurface 现场菜单子选项的回调函数
       case 'isotogglevisible'
        [a s]=getarrowslice;
%修改slice切面的可视性属性
        if propcheck(s,'visible','on')
          set(s,'visible','off');
        else
          set(s,'visible','on');
        end
       case 'isodelete'
%删除iso对象
        [a s]=getarrowslice;
        if numel(a)==1
          delete(getappdata(a,'arrowedge'));
        end
        cap=getappdata(s,'sliceomaticisocap');
        if ~isempty(cap)
          delete(cap);
        end
        delete(s);
        delete(a);
%设置isosurface对象的光照属性
       case 'isoflatlight'
        [a s]=getarrowslice;
        set(s,'facelighting','flat');
       case 'isosmoothlight'
        [a s]=getarrowslice;
        set(s,'facelighting','phong');
%设置isosurface对象的颜色属性
       case 'isocolor'
        [a s]=getarrowslice;
        c=uisetcolor(get(s,'facecolor'));
        slowset(s,'facecolor',c,d.animincrement);
%设置isosurface对象的透明属性
       case 'isoalpha'
        [a s]=getarrowslice;
        if nargin ~= 2
          error('Not enough arguments to sliceomatic.');
        end
        slowset(s,'facealpha',eval(p2),d.animincrement);
%设置isocap对象的属性
       case 'isocaps'
        [a s]=getarrowslice;
        cap=getappdata(s,'isosurfacecap');
        if isempty(cap)
          new=localisocaps(s);
          set(new,'uicontextmenu',d.uiciso);
        else
          delete(cap);
          setappdata(s,'isosurfacecap',[]);
        end
```

```
            % slice现场菜单子选项
        case 'togglevisible'
         [a s]=getarrowslice;
         switch get(s,'visible')
          case 'on'
           set(s,'visible','off');
           pushset(a,'facealpha',.2);
          case 'off'
           set(s,'visible','on');
           popset(a,'facealpha');
         end
% 设置slice对象的颜色属性
        case 'setfaceted'
         [a s]=getarrowslice;
         set(s,'edgec','k','facec','flat');
        if ischar(get(s,'facea')) && strcmp(get(s,'facea'),'texturemap')
           set(s,'facea','flat');
         end
         textureizeslice(s,'off');
        case 'setflat'
         [a s]=getarrowslice;
         set(s,'edgec','n','facec','flat');
        if ischar(get(s,'facea')) && strcmp(get(s,'facea'),'texturemap')
           set(s,'facea','flat');
         end
         textureizeslice(s,'off');
        case 'setinterp'
         [a s]=getarrowslice;
         set(s,'edgec','n','facec','interp');
        if ischar(get(s,'facea')) && strcmp(get(s,'facea'),'texturemap')
           set(s,'facea','interp');
         end
         textureizeslice(s,'off');
% 设置slice对象的材质属性
        case 'settexture'
         [a s]=getarrowslice;
         set(s,'facecolor','texture','edgec','none');
         if ischar(get(s,'facea'))
           set(s,'facealpha','texturemap');
         end
         textureizeslice(s,'on');
        case 'setnone'
         [a s]=getarrowslice;
         set(s,'facecolor','none','edgec','none');
         textureizeslice(s,'off');
% 设置slice对象的透明属性
        case 'setalphanone'
         [a s]=getarrowslice;
         slowset(s,'facealpha',1,d.animincrement);
        case 'setalphapoint5'
         [a s]=getarrowslice;
         slowset(s,'facealpha',.5,d.animincrement);
        case 'setalphaflat'
         [a s]=getarrowslice;
         set(s,'facealpha','flat');
        if ischar(get(s,'facec')) && strcmp(get(s,'facec'),'texturemap')
```

```
            set(s,'facecolor','flat');
            textureizeslice(s,'off');
          end
        case 'setalphainterp'
          [a s]=getarrowslice;
          set(s,'facealpha','interp');
        if ischar(get(s,'facec')) && strcmp(get(s,'facec'),'texturemap')
            set(s,'facecolor','interp');
            textureizeslice(s,'off');
          end
        case 'setalphatexture'
          [a s]=getarrowslice;
          set(s,'facealpha','texturemap');
          if ischar(get(s,'facec'))
            set(s,'facecolor','texturemap');
            textureizeslice(s,'on');
          end
%设置slice对象的等高线属性
        case 'slicecontour'
          [a s]=getarrowslice;
          localcontour(s, getappdata(s,'contour'));
        case 'slicecontourfullauto'
          [a s]=getarrowslice;
          d = getappdata(gcf, 'sliceomatic');
          minmax = get(d.axiso,'clim');
          levels = minmax(1):(minmax(2)-minmax(1))/10:minmax(2);
          setappdata(s, 'contourlevels', levels);
          localcontour(s, getappdata(s,'contour'),levels);
        case 'slicecontour_setauto'
          [a s]=getarrowslice;
          setappdata(s, 'contourlevels', []);
          localcontour(s, getappdata(s,'contour'));
%设置slice对象的等高线属性，包含等高线的高度属性
        case 'slicecontour_setfullauto'
          [a s]=getarrowslice;
          minmax = get(d.axiso,'clim');
          levels = minmax(1):(minmax(2)-minmax(1))/10:minmax(2);
          setappdata(s, 'contourlevels', levels);
          localcontour(s, getappdata(s,'contour'),levels);
        case 'slicecontour_select'
          [a s]=getarrowslice;
          d = getappdata(gcf, 'sliceomatic');
          xl = get(d.axiso,'xlim');
          levels = selectcontourlevels(get(s,'cdata'), xl(1), xl(2));
          setappdata(s, 'contourlevels', levels);
          localcontour(s, getappdata(s,'contour'),levels);
        case 'slicecontour_setlevels'
          [a s]=getarrowslice;
          d = getappdata(gcf, 'sliceomatic');
          xl = get(d.axiso,'xlim');
          levels = selectcontourlevels(get(s,'cdata'), xl(1), xl(2));
          setappdata(s, 'contourlevels', levels);
          localcontour(s, getappdata(s,'contour'),levels);
%删除slice对象的子菜单选项
        case 'deleteslice'
          [a s]=getarrowslice;
```

```
        if prod(size(a))==1
          delete(getappdata(a,'arrowedge'));
        end
        if ~isempty(getappdata(s,'contour'))
          delete(getappdata(s,'contour'));
        end
        delete(s);
        delete(a);
% 删除 slice 对象的等高线子菜单选项
      case 'deleteslicecontour'
        [a s]=getarrowslice;
        if ~isempty(getappdata(s,'contour'))
          delete(getappdata(s,'contour'));
        end
        temp=getappdata(s);
        try
          temp.contourlevels;
          setappdata(s,'contourlevels',[]);
        end
        setappdata(s,'contour',[]);
% 设置 slice 对象的等高线颜色属性
      case 'slicecontourflat'
        [a s]=getarrowslice;
        c = getappdata(s,'contour');
        if ~isempty(c)
          set(c,'edgecolor','flat');
        end
      case 'slicecontourinterp'
        [a s]=getarrowslice;
        c = getappdata(s,'contour');
        if ~isempty(c)
          set(c,'edgecolor','interp');
        end
      case 'slicecontourblack'
        [a s]=getarrowslice;
        c = getappdata(s,'contour');
        if ~isempty(c)
          set(c,'edgecolor','black');
        end
      case 'slicecontourwhite'
        [a s]=getarrowslice;
        c = getappdata(s,'contour');
        if ~isempty(c)
          set(c,'edgecolor','white');
        end
% 设置 slice 对象的等高线直线属性
      case 'slicecontoursmooth'
        [a s]=getarrowslice;
        c = getappdata(s,'contour');
        onoff = get(gcbo,'checked');
        switch onoff
         case 'off'
          set(c,'linesmoothing','on');
         case 'on'
          set(c,'linesmoothing','off');
        end
```

```
case 'slicecontourcolor'
 [a s]=getarrowslice;
 c = getappdata(s,'contour');
 if ~isempty(c)
   inputcolor = get(c,'edgecolor');
   if isa(inputcolor,'char')
     inputcolor=[ 1 1 1 ];
   end
  slowset(c,'edgecolor',uisetcolor(inputcolor),d.animincrement);
 end
case 'slicecontourlinewidth'
 [a s]=getarrowslice;
 c = getappdata(s,'contour');
 if ~isempty(c)
   if isa(p2,'char')
     slowset(c,'linewidth',str2num(p2),d.animincrement);
   else
     slowset(c,'linewidth',p2,d.animincrement);
   end
 end
 %
 % All Slices菜单的回调函数
 %
case 'allfacet'
 s=allSlices;
 set(s,'facec','flat','edgec','k');
 textureizeslice(s,'off');
case 'allflat'
 s=allSlices;
 set(s,'facec','flat','edgec','none');
 textureizeslice(s,'off');
case 'allinterp'
 s=allSlices;
 set(s,'facec','interp','edgec','none');
 textureizeslice(s,'off');
case 'alltex'
 s=allSlices;
 set(s,'facec','texturemap','edgec','none');
 textureizeslice(s,'on');
case 'allnone'
 s=allSlices;
 set(s,'facec','none','edgec','none');
 textureizeslice(s,'off');
case 'alltnone'
 s=allSlices;
 set(s,'facea',1);
 textureizeslice(s,'off');
case 'alltp5'
 s=allSlices;
 set(s,'facea',.5);
 textureizeslice(s,'off');
case 'alltflat'
 s=allSlices;
 set(s,'facea','flat');
 textureizeslice(s,'off');
case 'alltinterp'
```

```
 s=allSlices;
 set(s,'facea','interp');
 textureizeslice(s,'off');
case 'alltex'
 s=allSlices;
 set(s,'facea','texturemap');
 textureizeslice(s,'on');
 % 设置菜单的默认属性
 % 设置照明菜单的默认属性
case    'defaultfaceted'
 d.defcolor='faceted';
case    'defaultflat'
 d.defcolor='flat';
case    'defaultinterp'
 d.defcolor='interp';
case    'defaulttexture'
 d.defcolor='texture';
 if strcmp(d.defalpha,'flat') || strcmp(d.defalpha,'interp')
   d.defalpha='texture';
 end
case    'defaultinterp'
 d.defcolor='none';
 % 设置透明菜单的默认属性
case    'defaulttransnone'
 d.defalpha='none';
case    'defaulttransflat'
 d.defalpha='flat';
case    'defaulttransinterp'
 d.defalpha='interp';
case    'defaulttranstexture'
 d.defalpha='texture';
 d.defcolor='texture';
 % 设置光照菜单的默认属性
case       'defaultlightflat'
 d.deflight='flat';
case  'defaultlightsmooth'
 d.deflight='smooth';
 % 设置等高线菜单的默认属性
case 'defaultcontoursmooth'
 d.defaultcontoursmooth='on';
case 'defaultcontourflat'
 d.defcontourcolor='flat';
case 'defaultcontourinterp'
 d.defcontourcolor='interp';
case 'defaultcontourblack'
 d.defcontourcolor='black';
case 'defaultcontourwhite'
 d.defcontourcolor='white';
case 'defaultcontourlinewidth'
 if isa(p2,'char')
   d.defcontourlinewidth=str2num(p2);
 else
   d.defcontourlinewidth=p2;
 end
 % 显示camera工具栏
case 'cameratoolbar'
```

```
 cameratoolbar('Toggle');
case 'annotationtoolbar'
 if propcheck(d.toolbar,'visible','on')
   set(d.toolbar,'vis','off');
 else
   set(d.toolbar,'vis','on');
 end
 % 控件属性
case 'controlalpha'
 val=str2num(p2);
 iso=findobj(d.axiso,'type','image');
 if val == 0
   set([ d.pxx d.pxy d.pxz iso],'visible','off');
 else
   set([ d.pxx d.pxy d.pxz iso],'visible','on');
  slowset([ d.pxx d.pxy d.pxz] , 'facealpha',val,d.animincrement);
   slowset(iso,'alphadata',val,d.animincrement);
 end
case 'toggleanimation'
 if d.animincrement == 0
   d.animincrement = 10;
 else
   d.animincrement = 0;
 end
case 'controllabels'
 l = get(d.axx,'xticklabel');
 if isempty(l)
   set([ d.axx d.axiso],'xticklabelmode','auto');
   set([ d.axy d.axz],'yticklabelmode','auto');
 else
   set([ d.axx d.axiso],'xticklabel',[]);
   set([ d.axy d.axz],'yticklabel',[]);
 end
case 'controlvisible'
 objs=findobj([ d.axiso d.axx d.axy d.axz]);
 if strcmp(get(d.axx,'visible'),'on')
   set(objs,'visible','off');
   set(d.axmain,'pos',[ .1 .1 .9 .8]);
 else
   set(objs,'visible','on');
  set(d.axmain,'pos',[ .2  .2 .6 .6]);
 end
 %
 % UICONTROL 回调函数
 %
case 'colormap'
 str=get(gcbo,'string');
 val=str{get(gcbo,'value')};
 size(val);
 if strcmp(val,'custom')
   cmapeditor
 else
   slowset(gcf,'colormap',feval(val),d.animincrement);
 end
case 'alphamap'
 str=get(gcbo,'string');
```

```
        val=alphamap(str(get(gcbo,'value')));
        [illegible]
        % Commands
        [illegible]
        copyobj(gca,figure);set(gca,'pos',[.1 .1 .9 .8]);
        [illegible]
        newf=[illegible],'[illegible]','[illegible]',get(gcf,'[illegible]');
        copyobj(d.axmain,newf);
        set(gca,'pos',[.1 .1 .9 .8])
        printdig(newf);
        close(newf);
        [illegible]
        [illegible]
      end
    [illegible]
      disp(get(0,'errormessage'));
    end
    [illegible]
  else
    disp('Sliceomatic data must be DOUBLE');
  end
```

上面程序是[illegible]的，[illegible]，结构[illegible]，其为[illegible]format setdata 函数，[illegible] Sliceomatic.fig 命令[illegible]，[illegible]，根据 Sliceomatic.fig 中各个参数[illegible]取值，调用[illegible]的函数命令，[illegible]，[illegible]在输入量的 case 和[illegible] else 和 end 循环[illegible]结构。

[illegible]

1198 设置 GUI 对象的菜单选项

继续上面小节的步骤。

step 1 在上面步骤打开的 M 文件编辑器中，输入下面的代码

```
function fileName = UserStickyPrefsFileName
localPath = fileparts(which(mfilename));
fileName = fullfile(localPath,'Sliceomatic.Prefs.mat');
```

[illegible]，[illegible]，[illegible]，然后使用 fullfile 命令返回完整的文件名称。

[illegible]

step 2 在上面步骤打开的 M 文件编辑器中，输入下面的代码

```
%-----------------------------------------------------------------------
function dOut = OverrideStickyUserPreferences(d)
%characteristic prefs file (stored locally where Slice-O-Matic installed)
fileName = UserStickyPrefsFileName;
%加载属性数值
if exist(fileName,'file')
  load(fileName)
  set(d.toolbar,'visible',prefs.anntoolbar_Checked)
  if prefs.camtoolbar_checked
    cameratoolbar('show');
  else
    cameratoolbar('hide');
  end
%读取对应的属性数值
  d.defcolor = prefs.defcolor;
  d.defalpha = prefs.defalpha;
  d.deflight = prefs.deflight;
  d.defcontourcolor = prefs.defcontourcolor;
  d.defcontourlinewidth = prefs.defcontourlinewidth;
  d.defcontoursmooth = prefs.defcontoursmooth;
%判断图形的坐标轴标签属性模式
    if strcmp('auto',prefs.ticklabels)
    set([d.axx d.axiso],'xticklabelmode','auto');
    set([d.axy d.axz],'yticklabelmode','auto');
  else
    set([d.axx d.axiso],'xticklabel',[]);
    set([d.axy d.axz],'yticklabel',[]);
  end
  d.animincrement = prefs.animincrement;
  set([d.pxx d.pxy d.pxz] , 'facealpha',prefs.controlalpha);
  iso = findobj(d.axiso,'type','image');
  set(iso,'alphadata',prefs.controlalpha);
%显示已经加载属性列表数值
  disp('Sticky preferences loaded.')
end
%返回编辑后的图形属性结构体变量
dOut = d;
```

上面的程序代码功能是覆盖用户自行设置的对象属性，在程序代码的开始，首先加载文件中的数据，然后设置图形界面中的对象属性。

step 3 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
%-----------------------------------------------------------------------
function SavePrefs(obj,event)
%保存关于对象属性的结构体
d = getappdata(gcf,'sliceomatic');
%extract only preferences that need to be sticky
prefs.anntoolbar_Checked = get(d.toolbar,'Visible');
prefs.defcolor = d.defcolor;
prefs.defalpha = d.defalpha;
prefs.deflight = d.deflight;
prefs.defcontourcolor = d.defcontourcolor;
prefs.defcontourlinewidth = d.defcontourlinewidth;
prefs.defcontoursmooth = d.defcontoursmooth;
prefs.camtoolbar_checked = cameratoolbar('getvisible');
```

```
prefs.ticklabels = get(d.axx,'xticklabelmode');
prefs.animincrement = d.animincrement;
prefs.controlalpha = get(d.pxx,'facealpha');
%将属性列表数值保存到对应的文件中
fileName = UserStickyPrefsFileName;
save(fileName,'prefs')
disp([ 'Saved: ' fileName])
```

上面的程序代码设置的是该GUI对象保存文件的属性，当在程序代码中调用该程序时，将会直接设置所有的保存信息。

step 4 在上面步骤中打开的M文件编辑器中，输入下面的代码：

```
function controlmenu(fig, action)
%处理关于控制菜单的程序代码
   d=getappdata(gcf,'sliceomatic');
  if cameratoolbar('getvisible')
    set(d.camtoolbar,'checked','on');
  else
    set(d.camtoolbar,'checked','off');
  end
    if exist('uitoolfactory') == 2
    if propcheck(d.toolbar,'visible','on')
      set(d.anntoolbar,'checked','on');
    else
      set(d.anntoolbar,'checked','off');
    end
  end
     set([ d.dcalpha1 d.dcalpha8 d.dcalpha6 d.dcalpha5 d.dcalpha6 d.
dcalpha2 d.dcalpha0...
       d.dclabels d.dcvis ],...
      'checked','off');
  switch get(d.pxx,'facealpha')
   case 1, set(d.dcalpha1,'checked','on');
   case .8, set(d.dcalpha8,'checked','on');
   case .6, set(d.dcalpha6,'checked','on');
   case .5, set(d.dcalpha5,'checked','on');
   case .4, set(d.dcalpha4,'checked','on');
   case .2, set(d.dcalpha2,'checked','on');
   case 0, set(d.dcalpha0,'checked','on');
  end
  if d.animincrement == 0
    set(d.dcanimstep,'checked','off');
  else
    set(d.dcanimstep,'checked','on');
  end
    if ~isempty(get(d.axx,'xticklabel'))
    set(d.dclabels,'checked','on');
  end
    if strcmp(get(d.axx,'visible'),'on')
    set(d.dcvis,'checked','on');
  end
    if 0
    xt = get(get(d.axx,'title'),'string');
    switch xt
     case 'X Slice Controller'
```

```
      set(d.dcslice,'checked','on');
    end
    xt = get(get(d.axiso,'title'),'string');
    switch xt
     case 'Iso Surface Controller'
      set(d.dciso,'checked','on');
    end
  end
```

上面程序代码的主要功能是设置"Controls"菜单选项的各种属性，首先设置照明工具栏和注释工具栏中的按钮的检录属性"checked"的数值。在MATLAB中，菜单选项的检录属性的取值有on和off两个。默认情况下，菜单选项的"checked"属性数值为"off"，不会显示检录符，当其属性的数值为"on"时，用户选中该菜单选项，其选项就会出现"√"标记。在本实例中，"Controls"菜单选项的主要功能是控制工具栏的显示选项和图形对象的控制选项（透明度设置）等。

step 5 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
function defaultmenu(fig, action)
% Handle toggling bits on the slice defaults menu
    d=getappdata(gcf,'sliceomatic');
    set([ d.dfacet d.dflat d.dinterp d.dtex d.dtnone d.dtflat d.dtinterp ...
        d.dttex d.dcflat d.dcinterp d.dcblack d.dcwhite d.dcnone ...
        d.dlflat d.dlsmooth ...
       d.smcl1 d.smcl2 d.smcl3 d.smcl4 d.smcl5 d.smcl6 ], 'checked','off');
  switch d.defcolor
   case 'faceted'
    set(d.dfacet,'checked','on');
   case 'flat'
    set(d.dflat,'checked','on');
   case 'interp'
    set(d.dinterp,'checked','on');
   case 'texture'
    set(d.dtex,'checked','on');
   case 'none'
    set(d.dcnone,'checked','on');
  end
  switch d.defalpha
   case 'none'
    set(d.dtnone,'checked','on');
   case 'flat'
    set(d.dtflat,'checked','on');
   case 'interp'
    set(d.dtinterp,'checked','on');
   case 'texture'
    set(d.dttex,'checked','on');
  end
  switch d.deflight
   case 'flat'
    set(d.dlflat,'checked','on');
   case 'smooth'
    set(d.dlsmooth,'checked','on');
  end
  switch d.defcontourcolor
```

```
 case 'flat'
  set(d.dcflat,'checked','on');
 case 'interp'
  set(d.dcinterp,'checked','on');
 case 'black'
  set(d.dcblack,'checked','on');
 case 'white'
  set(d.dcwhite,'checked','on');
end
%set(d.dcsmooth,'checked',d.defcontoursmooth);
switch d.defcontourlinewidth
 case 1, set(d.dcl1,'checked','on');
 case 2, set(d.dcl2,'checked','on');
 case 3, set(d.dcl3,'checked','on');
 case 4, set(d.dcl4,'checked','on');
 case 5, set(d.dcl5,'checked','on');
 case 6, set(d.dcl6,'checked','on');
end
```

上面程序代码的主要功能是设置“Object_Defaults”菜单选项的各种属性。在本GUI实例中，“Object_Defaults”菜单选项提供用户设置Slice截面的颜色、透明度，Isosurface曲面的光照和等高线的颜色等属性。用户可以选择对应的菜单选项，设置这些图形对象的默认属性。当用户选中对应选项后，其选项就会出现“√”标记。

step 6 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
function slicecontextmenu(fig,action)
% Context menu state for slices
  d=getappdata(gcf,'slicematic');
  [a s]=getarrowslice;
  set([d.smfacet d.smflat d.sminterp d.smtex d.smtnone d.smtp5 ...
       d.smtflat d.smtinterp d.smttex d.smnone d.smcsmooth
      ],'checked','off');
  set(d.vistog,'checked',get(s,'visible'));
    if propcheck(s,'edgec',[0 0 0])
    set(d.smfacet,'checked','on');
  elseif propcheck(s,'facec','flat')
    set(d.smflat,'checked','on');
  end
  if propcheck(s,'facec','interp')
    set(d.sminterp,'checked','on');
  end
  if propcheck(s,'facec','texturemap')
    set(d.smtex,'checked','on');
  end
  if propcheck(s,'facec','none')
    set(d.smnone,'checked','on');
  end
  if propcheck(s,'facea',1)
    set(d.smtnone,'checked','on');
  end
  if propcheck(s,'facea',.5)
    set(d.smtp5,'checked','on');
  end
  if propcheck(s,'facea','flat')
```

```
  set(d.smtflat,'checked','on');
end
if propcheck(s,'facea','interp')
  set(d.smtinterp,'checked','on');
end
if propcheck(s,'facea','texturemap')
  set(d.smttex,'checked','on');
end
cm = [ d.smcflat d.smcinterp d.smcblack d.smcwhite d.smccolor ...
     d.smcl1 d.smcl2 d.smcl3 d.smcl4 d.smcl5 d.smcl6 ];
set(cm,'checked','off');
if isempty(getappdata(s,'contour'))
  set(d.smcontour,'enable','on');
  set(d.smcsetauto,'enable','off');
  set(d.smcsetav,'enable','off');
  set(d.smclevels,'enable','off');
  set(d.smrcontour,'enable','off');
  set(d.smcsmooth,'enable','off');
  set(cm,'enable','off');
else
  set(d.smcontour,'enable','off')
  set(d.smcsetauto,'enable','on');
  set(d.smcsetav,'enable','on');
  set(d.smclevels,'enable','on');
  set(d.smrcontour,'enable','on')
  set(d.smcsmooth,'enable','on');
  set(cm,'enable','on')
  c = getappdata(s,'contour');
  if propcheck(c,'linesmoothing','on')
    set(d.smcsmooth,'checked','on');
  end
  ec = get(c,'edgecolor');
  if isa(ec,'char')
    switch ec
     case 'flat'
      set(d.smcflat,'checked','on');
     case 'interp'
      set(d.smcinterp,'checked','on');
    end
  else
    if ec == [  1 1 1 ]
      set(d.smcwhite,'checked','on');
    elseif ec == [  0 0 0 ]
      set(d.smcblack,'checked','on');
    else
      set(d.smccolor,'checked','on');
    end
  end
  clw = get(c,'linewidth');
  switch clw
   case 1, set(d.smcl1,'checked','on');
   case 2, set(d.smcl2,'checked','on');
   case 3, set(d.smcl3,'checked','on');
   case 4, set(d.smcl4,'checked','on');
   case 5, set(d.smcl5,'checked','on');
   case 6, set(d.smcl6,'checked','on');
```

```
    end
  end
```

上面程序代码的主要功能是设置关于Slice属性的现场菜单选项，用户可以设置Slice截面的颜色、透明度、照明处理、线宽等属性。上面程序代码的结构并不复杂，请用户自行分析其代码含义。

step 7 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
function isocontextmenu(fig,action)
% Context menu state for isosurfaces
  d=getappdata(gcf,'sliceomatic');
    [a s]=getarrowslice;
  if propcheck(s,'facelighting','flat')
    set(d.isoflatlight,'checked','on');
    set(d.isosmoothlight,'checked','off');
  else
    set(d.isoflatlight,'checked','off');
    set(d.isosmoothlight,'checked','on');
  end
  set(d.vistogiso,'checked',get(s,'visible'));
  if ~isempty(getappdata(s,'isosurfacecap'))
    set(d.isocap,'checked','on');
  else
    set(d.isocap,'checked','off');
  end
```

上面程序代码的主要功能是设置关于Isosurface曲面属性的现场菜单选项，用户可以设置该曲面的光照属性等。

step 8 在上面步骤打开的M文件编辑器中，输入下面的代码：

```
function outd = figmenus(d)
% Set up sliceomatic's gui menus within structure D
% Main Figure Menu
  set(gcf,'menubar','none');

% File menu
  d.filemenu = uimenu(gcf,'label','File');
   d.fcopy = uimenu(d.filemenu, 'label', 'Copy figure','callback',
'sliceomatic copy');
   d.fprint  = uimenu(d.filemenu,'label','Print...','callback',
'sliceomatic print');
  %%% start patch 1of3 RAB 2/18/05 %%%
  d.fsaveprefs = uimenu(d.filemenu,'label','Save preferences','callback',
@SavePrefs);
  %%% end patch 1of3 RAB 2/18/05 %%%
  % How do get these props onto the print figure?
  %d.fprints = uimenu(d.filemenu,'label','Print Setup...','callback',
'printdlg -setup');
  % ---
  d.fexit = uimenu(d.filemenu, 'label', 'Close','callback','closereq',...
                   'separator','on');
% Controls Menu
   d.defcontrols = uimenu(gcf,'label','Controls', 'callback',
```

```
@controlmenu);
  if exist('uitoolfactory') == 2
    d.anntoolbar = uimenu(d.defcontrols,'label','Annotations toolbar',
'callback', 'sliceomatic annotationtoolbar');
  end
   d.camtoolbar = uimenu(d.defcontrols,'label','Camera toolbar',
'callback', 'sliceomatic cameratoolbar');
  d.dcalpha = uimenu(d.defcontrols,'label','Controls Transparency');
  d.dcalpha1= uimenu(d.dcalpha,'label','1','callback','sliceomatic
controlalpha 1');
  d.dcalpha8= uimenu(d.dcalpha,'label','.8','callback','sliceomatic
controlalpha .8');
  d.dcalpha6= uimenu(d.dcalpha,'label','.6','callback','sliceomatic
controlalpha .6');
  d.dcalpha5= uimenu(d.dcalpha,'label','.5','callback','sliceomatic
controlalpha .5');
  d.dcalpha4= uimenu(d.dcalpha,'label','.4','callback','sliceomatic
controlalpha .4');
  d.dcalpha2= uimenu(d.dcalpha,'label','.2','callback','sliceomatic
controlalpha .2');
  d.dcalpha0= uimenu(d.dcalpha,'label','0','callback','sliceomatic
controlalpha 0');
  d.dcanimstep = uimenu(d.defcontrols,'label','Animation','callback',
'sliceomatic toggleanimation');
  d.dclabels= uimenu(d.defcontrols','label','Tick Labels','callback',
'sliceomatic controllabels');
   d.dcvis    = uimenu(d.defcontrols','label','Visible','callback',
'sliceomatic controlvisible');
  % d.dsetrange= uimenu(d.defcontrols','label','Set Range','callback',
'@setvolumerange');
  % d.dcslice = uimenu(d.defcontrols,'label','Slice Controls','callback',
'sliceomatic useslicecontrols');
   % d.dciso    = uimenu(d.defcontrols,'label','Iso Surface Control',
'callback','sliceomatic useisocontrols','separator','on');

   % Remove this once we have more controls to enable and disable.
   %  set(d.defcontrols,'vis','off');

   % Default for new slices menu
   d.defmenu = uimenu(gcf,'label','Object_Defaults', 'callback',
@defaultmenu);
  d.dfacet  = uimenu(d.defmenu,'label','Slice Color Faceted','callback',
'sliceomatic defaultfaceted');
  d.dflat   = uimenu(d.defmenu,'label','Slice Color Flat',   'callback',
'sliceomatic defaultflat');
  d.dinterp = uimenu(d.defmenu,'label','Slice Color Interp', 'callback',
'sliceomatic defaultinterp');
  d.dtex    = uimenu(d.defmenu,'label','Slice Color Texture','callback',
'sliceomatic defaulttexture');
  d.dcnone  = uimenu(d.defmenu,'label','Slice Color None','callback',
'sliceomatic defaultcolornone');
   d.dtnone   = uimenu(d.defmenu,'label','Slice Transparency None',
'callback','sliceomatic defaulttransnone','separator','on');
   d.dtflat   = uimenu(d.defmenu,'label','Slice Transparency Flat',
'callback','sliceomatic defaulttransflat');
   d.dtinterp= uimenu(d.defmenu,'label','Slice Transparency Interp',
```

```
'callback','sliceomatic defaulttransinterp');
  d.dttex   = uimenu(d.defmenu,'label','Slice Transparency Texture',
'callback','sliceomatic defaulttranstexture');
  d.dlflat  = uimenu(d.defmenu,'label','IsoSurface Lighting Flat',
'callback','sliceomatic defaultlightflat','separator','on');
  d.dlsmooth= uimenu(d.defmenu,'label','IsoSurface Lighting Smooth',
'callback','sliceomatic defaultlightsmooth');
  %d.dcsmooth= uimenu(d.defmenu,'label','Contour Line Smoothing',
'callback','sliceomatic defaultcontoursmooth');
 d.dcflat  = uimenu(d.defmenu,'label','Contour Color Flat',  'callback',
'sliceomatic defaultcontourflat','separator','on');
  d.dcinterp= uimenu(d.defmenu,'label','Contour Color Interp',
'callback','sliceomatic defaultcontourinterp');
 d.dcblack = uimenu(d.defmenu,'label','Contour Color Black',  'callback',
'sliceomatic defaultcontourblack');
 d.dcwhite = uimenu(d.defmenu,'label','Contour Color White',  'callback',
'sliceomatic defaultcontourwhite');
  d.dclinew = uimenu(d.defmenu,'label','Contour Line Width');
  d.dcl1    = uimenu(d.dclinew,'label','1','callback','sliceomatic
defaultcontourlinewidth 1');
  d.dcl2    = uimenu(d.dclinew,'label','2','callback','sliceomatic
defaultcontourlinewidth 2');
  d.dcl3    = uimenu(d.dclinew,'label','3','callback','sliceomatic
defaultcontourlinewidth 3');
  d.dcl4    = uimenu(d.dclinew,'label','4','callback','sliceomatic
defaultcontourlinewidth 4');
  d.dcl5    = uimenu(d.dclinew,'label','5','callback','sliceomatic
defaultcontourlinewidth 5');
  d.dcl6    = uimenu(d.dclinew,'label','6','callback','sliceomatic
defaultcontourlinewidth 6');
    d.defcolor='texture';
  d.defalpha='texture';
  d.deflight='smooth';
  d.defcontourcolor='black';
  d.defcontourlinewidth=1;
  % This exposes an unpleasant R14 bug
  d.defcontoursmooth='off';
  % investigate hardware opengl.
  inc = 0;
  try
    od = opengl('data');
    if isfield(od,'Software')
      % R14 version of MATLAB
      if ~od.Software
        inc = 10;
      end
    else
      % Older version of MATLAB
      if ~(strcmp(od.Renderer,'Mesa X11') |  ...
           strcmp(od.Renderer, 'GDI Generic'))
        inc = 10;
      end
    end
  end
  d.animincrement=inc;
    %%% start patch 2of3 RAB 2/18/05 %%%
```

```
  d = OverrideStickyUserPreferences(d);
  %%% end patch 2of3 RAB 2/18/05 %%%
    % Set props for all slices menu
  d.allmenu = uimenu(gcf,'label','AllSlices');
  uimenu(d.allmenu,'label','Color Faceted','callback','sliceomatic
allfacet');
  uimenu(d.allmenu,'label','Color Flat','callback','sliceomatic
allflat');
  uimenu(d.allmenu,'label','Color Interp','callback','sliceomatic
allinterp');
  uimenu(d.allmenu,'label','Color Texture','callback','sliceomatic
alltex');
  uimenu(d.allmenu,'label','Color None','callback','sliceomatic
allnone');
  uimenu(d.allmenu,'label','Transparency None','callback','sliceomatic
alltnone','separator','on');
  uimenu(d.allmenu,'label','Transparency .5','callback','sliceomatic
alltp5');
  uimenu(d.allmenu,'label','Transparency Flat','callback','sliceomatic
alltflat');
  uimenu(d.allmenu,'label','Transparency Interp','callback','sliceomatic
alltinterp');
  uimenu(d.allmenu,'label','Transparency Texture','callback','sliceomatic
allttex');
  % Setup Help style options
  d.helpmenu = uimenu(gcf,'label','Help');
  uimenu(d.helpmenu,'label','Help','callback','doc sliceomatic/
sliceomatic');
  uimenu(d.helpmenu,'label','Check for Updates','callback','web http:/
/www.mathworks.com/matlabcentral/fileexchange/loadFile.do?
objectId=764&objectType=FILE');
  uimenu(d.helpmenu,'label','About Author','callback','web http://www.
mathworks.com/matlabcentral/fileexchange/loadAuthor.do?
objectId=803709&objectType=author');
  % Context Menus
  % Slice Context Menu
  d.uic=uicontextmenu('callback', @slicecontextmenu);
  d.vistog = uimenu(d.uic,'label','Visible','callback','sliceomatic
togglevisible');
  d.uicdelete = uimenu(d.uic,'label','Delete','callback','sliceomatic
deleteslice');
  d.smcolorm  = uimenu(d.uic,'label','Color','separator','on');
  d.smfacet   = uimenu(d.smcolorm,'label','Color Faceted','callback',
'sliceomatic setfaceted');
  d.smflat    = uimenu(d.smcolorm,'label','Color Flat','callback',
'sliceomatic setflat');
  d.sminterp  = uimenu(d.smcolorm,'label','Color Interp','callback',
'sliceomatic setinterp');
  d.smtex     = uimenu(d.smcolorm,'label','Color Texture','callback',
'sliceomatic settexture');
  d.smnone    = uimenu(d.smcolorm,'label','Color None','callback',
'sliceomatic setnone');
  d.smtransm  = uimenu(d.uic,'label','Transparency');
  d.smtnone   = uimenu(d.smtransm,'label','Transparency None','callback',
'sliceomatic setalphanone');
  d.smtp5     = uimenu(d.smtransm,'label','Transparency .5','callback',
```

```
'sliceomatic setalphapoint5');
 d.smtflat   = uimenu(d.smtransm,'label','Transparency Flat','callback',
'sliceomatic setalphaflat');
  d.smtinterp = uimenu(d.smtransm,'label','Transparency Interp',
'callback','sliceomatic setalphainterp');
  d.smttex     = uimenu(d.smtransm,'label','Transparency Texture',
'callback','sliceomatic setalphatexture');
 d.smcontour = uimenu(d.uic,'label','Add Contour','separator','on');
 d.smcont0   = uimenu(d.smcontour,'label','Auto (Slice)','callback',
'sliceomatic slicecontour');
 d.smcont0v  = uimenu(d.smcontour,'label','Auto (Volume)','callback',
'sliceomatic slicecontourfullauto');
 d.smcont1   = uimenu(d.smcontour,'label','Select Levels','callback',
'sliceomatic slicecontour select','separator','on');
  d.smcsetauto= uimenu(d.uic,'label','Set Auto Levels (Slice)',
'callback','sliceomatic slicecontour_setauto');
  d.smcsetav  = uimenu(d.uic,'label','Set Auto Levels (Volume)',
'callback','sliceomatic slicecontour_setfullauto');
 d.smclevels = uimenu(d.uic,'label','Set Levels','callback','sliceomatic
slicecontour_setlevels');
  d.smrcontour= uimenu(d.uic,'label','Remove Contour','callback',
'sliceomatic deleteslicecontour');
 d.smccm     = uimenu(d.uic,'label','Contour Colors');
 d.smcflat    = uimenu(d.smccm,'label','Contour Flat','callback',
'sliceomatic slicecontourflat');
 d.smcinterp = uimenu(d.smccm,'label','Contour Interp','callback',
'sliceomatic slicecontourinterp');
 d.smcblack  = uimenu(d.smccm,'label','Contour Black','callback',
'sliceomatic slicecontourblack');
 d.smcwhite  = uimenu(d.smccm,'label','Contour White','callback',
'sliceomatic slicecontourwhite');
 d.smccolor  = uimenu(d.smccm,'label','Contour Color','callback',
'sliceomatic slicecontourcolor');
 d.smcsmooth = uimenu(d.uic,'visible','off','label','Smooth Contour
Lines','callback','sliceomatic slicecontoursmooth');
 d.smclinew  = uimenu(d.uic,'label','Contour Line Width');
 d.smcl1     = uimenu(d.smclinew,'label','1','callback','sliceomatic
slicecontourlinewidth 1');
 d.smcl2     = uimenu(d.smclinew,'label','2','callback','sliceomatic
slicecontourlinewidth 2');
 d.smcl3     = uimenu(d.smclinew,'label','3','callback','sliceomatic
slicecontourlinewidth 3');
 d.smcl4     = uimenu(d.smclinew,'label','4','callback','sliceomatic
slicecontourlinewidth 4');
 d.smcl5     = uimenu(d.smclinew,'label','5','callback','sliceomatic
slicecontourlinewidth 5');
 d.smcl6     = uimenu(d.smclinew,'label','6','callback','sliceomatic
slicecontourlinewidth 6');
   % Isosurface Context Menu
 d.uiciso=uicontextmenu('callback',@isocontextmenu);
 d.vistogiso = uimenu(d.uiciso,'label','Visible','callback','sliceomatic
isotogglevisible');
 d.isodelete = uimenu(d.uiciso,'label','Delete','callback','sliceomatic
isodelete');
  d.isoflatlight=uimenu(d.uiciso,'label','LightingFlat','callback',
'sliceomatic oflatlight','separator','on');
```

```
[illegible]
  d.isocolor = uimenu(d.uiciso,'label','Change Color','callback',
[illegible]
isocaps','separator','on');
[illegible]
```

上面的[illegible]代码保存为“figmenus.m”文件。

11.9.9 检测程序代码

延续上面小节的步骤。

step 1 在MATLAB的命令窗口中输入下面的代码：

```
>> sliceomatic
```

step 2 查看命令窗口，输入[illegible]，按“Enter”键，得到下面的内容：

```
Smoothing for IsoNormals...
Generating reduction volume...
```

[illegible]MATLAB[illegible]默认[illegible]，如图11.104所示。

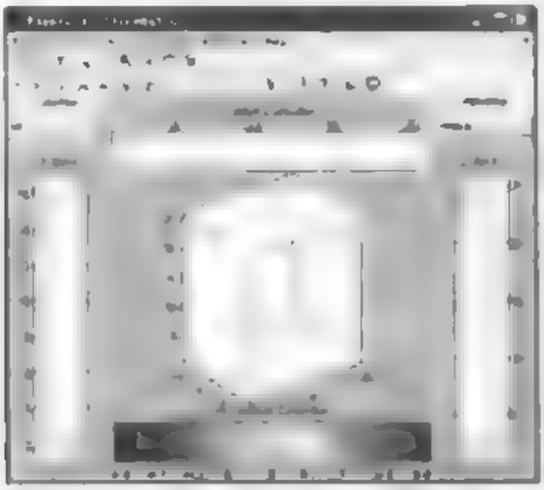


图11.104 默认的GUI图形界面

step 3 [illegible]坐标数值，得到的结果如图11.105所示。

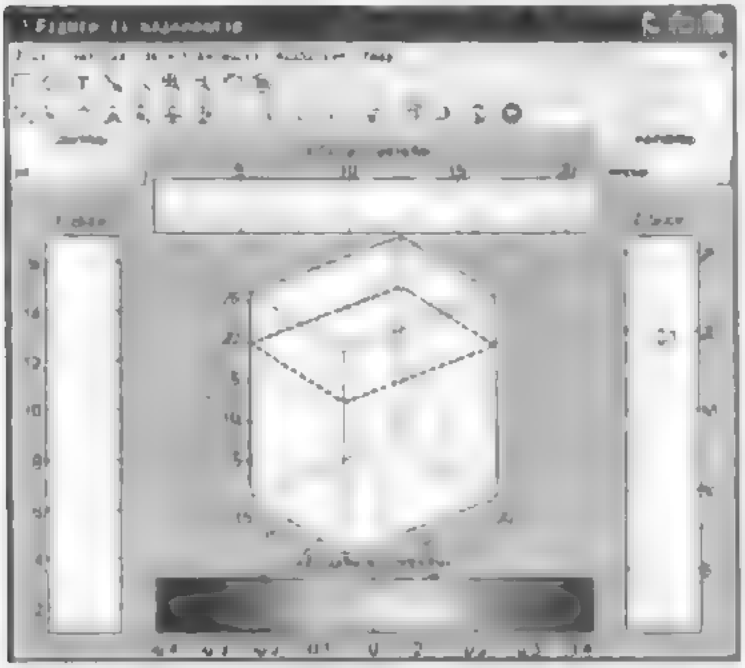


图 11.105 选择切面 Z 向坐标

[illegible]

step 4 [illegible]

[illegible]

step 5 [illegible]各个方向上的 Slice 截面，得到的结果如图 11.107 所示。

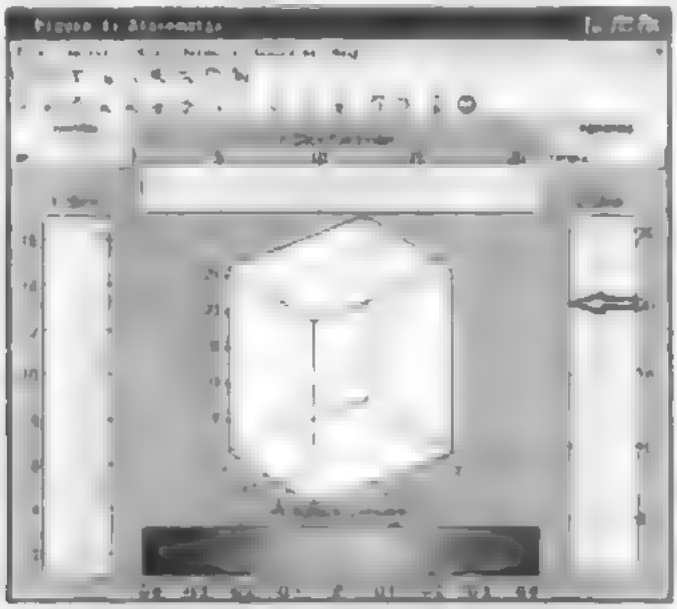


图 11.106 绘制 Slice 截面

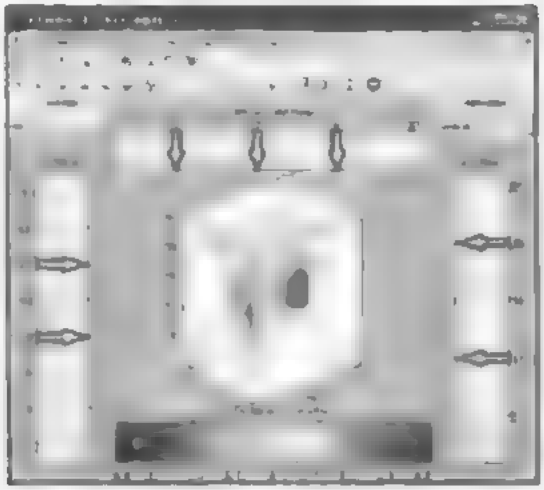


图 11.107　选择其他方向的截面

Step 6

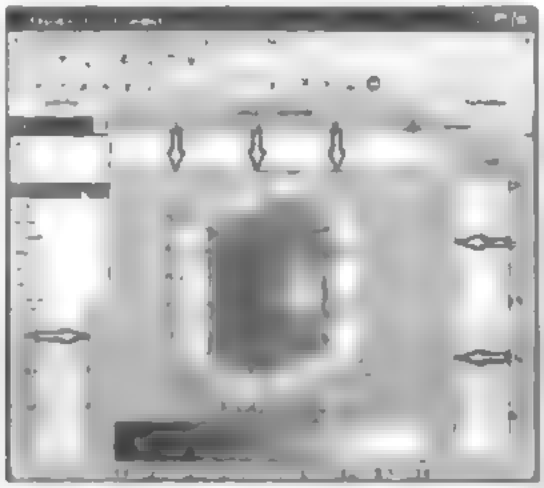


图 11.108　修改图形对象的色图颜色

Step 7

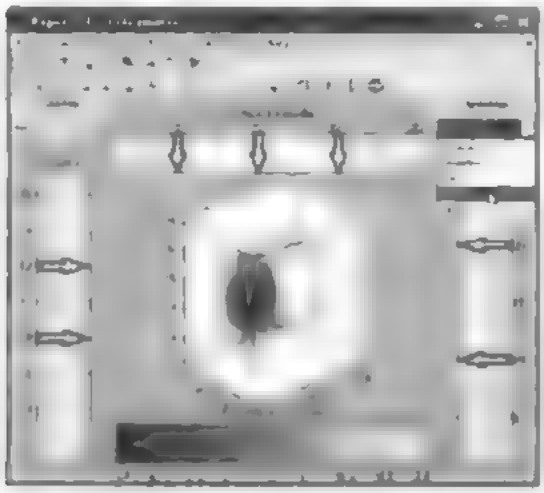


图 11.109　修改图形对象的透明度属性

step 8 [illegible]

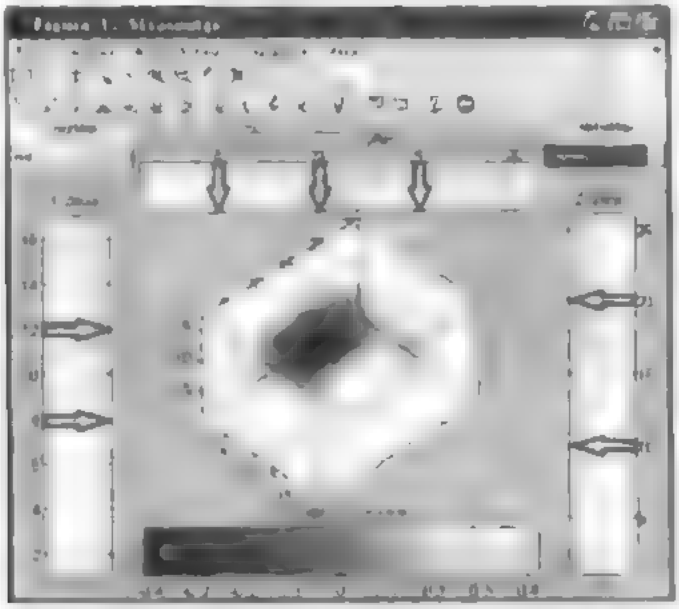


图 11.110 修改图形对象的视角

step 9 [illegible]

[illegible]

step 10 [illegible]

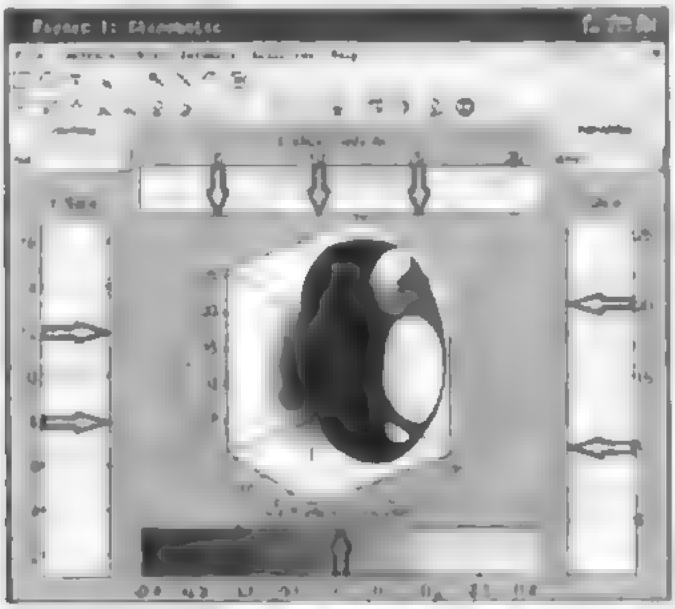


图 11.111 添加 isosurface 曲面

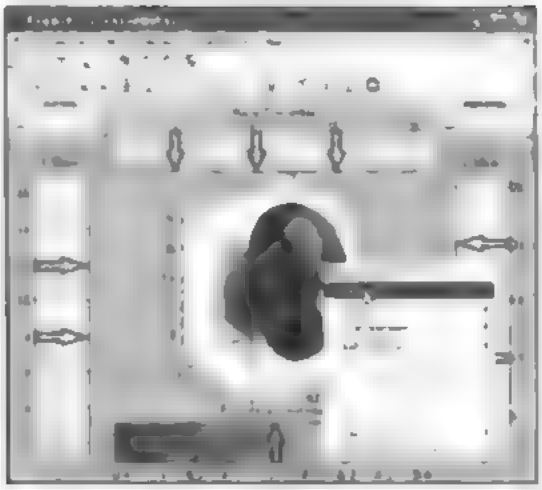


图 11.112　删除添加的 Isosurface 曲面

Step 11 [illegible]，如图 11.113 所示。

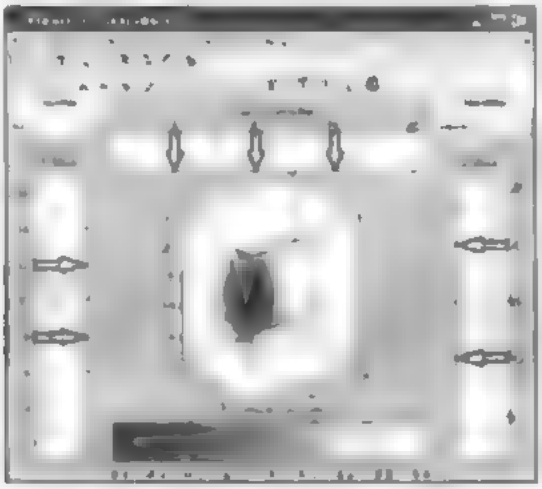


图 11.113　删除曲面后的图形对象

Step 12 [illegible] Slice 界面的颜色属性，得到的结果如图 11.114 所示。

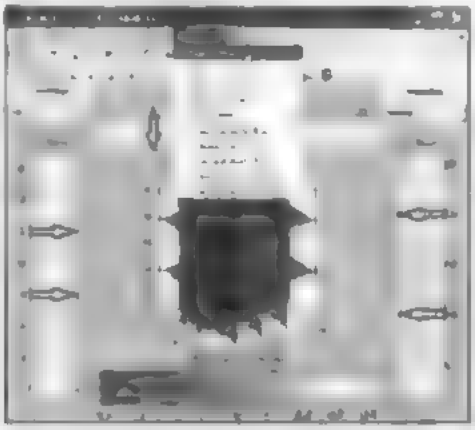


图 11.114　修改 Slice 截面的颜色属性

Step 13 [illegible]

项 "Add Contour" ⇨ "Auto Slice" 选项，如图 11.115 所示。

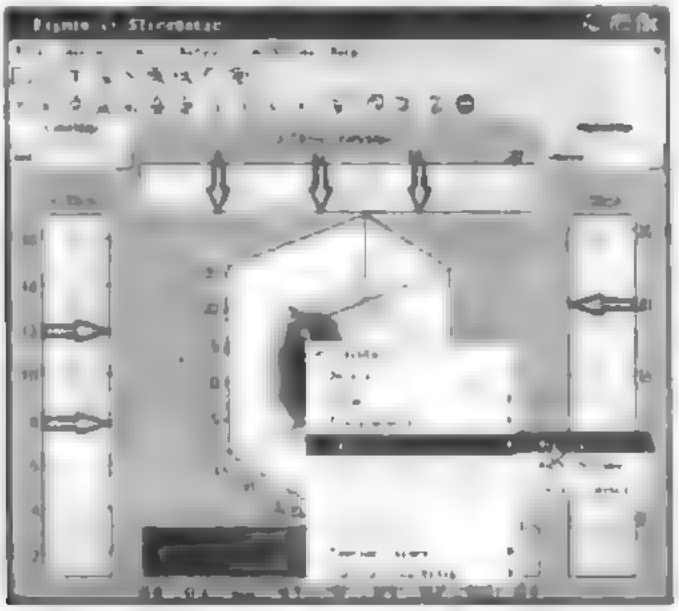


图 11.115 添加等高线

[illegible]

step 14 [illegible]

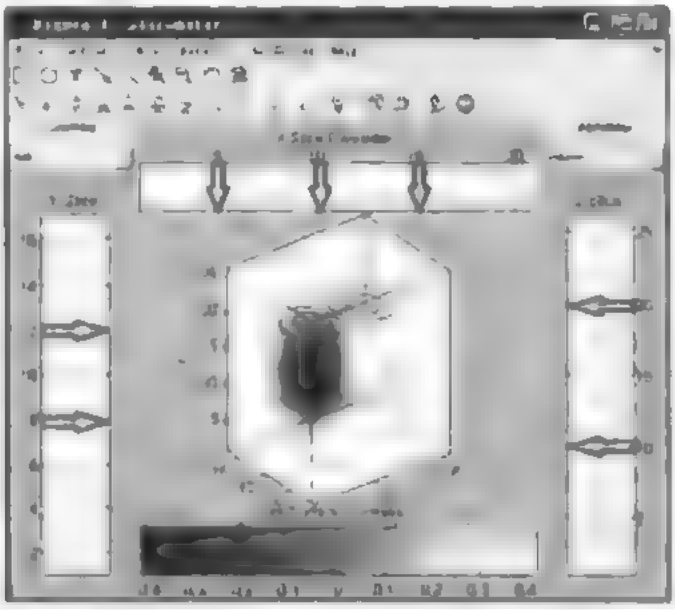


图 11.116 添加等高线后的图形界面

step 15 [illegible]

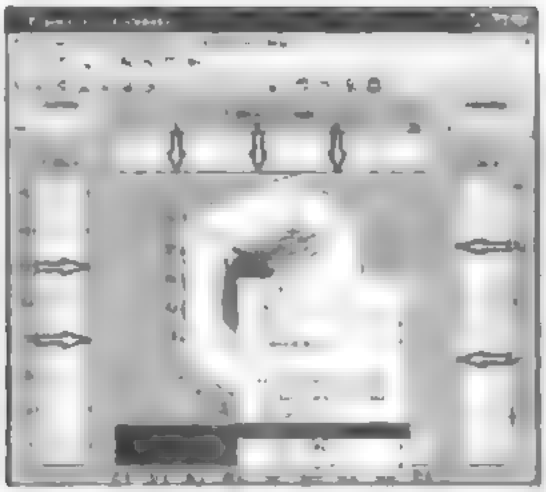


图 11.117 删除等高线

[illegible]

Step 16 [illegible] 令，打开一个空白的 M 文件编辑器，然后在 M 文件编辑器中输入下面[illegible]

```
function [centers,v,x,y,z]=
  Blinn's blobs
  [illegible]
  v=blob(centers,40,40,40)
  isosurface(v,.125)
  From ACM Transactions on Graphics, July 1982, Volume 1, Number 3.
  "A Generalization of A;gebraic Surface Drawing" James F. Blinn
    [illegible]
    [illegible];
    [illegible];
    a=.05;
    [illegible]
    [illegible];
    [illegible]
    for i=1:numCenters
        dx=centers(i,1)-x;
        dy=centers(i,2)-y;
        dz=centers(i,3)-z;
        v=v+b*exp(-a*(dx.^2 + dy.^2 + dz.^2));
    end
function x=makeXMat(nx,ny,nz)
    x=repmat([1:ny],[nx 1 nz]);
function y=makeYMat(nx,ny,nz)
    y=repmat([1:nx]',[1 ny nz]);
[illegible]
    z=repmat([1:nz],nx*ny,1);
z=reshape(z,[nx ny nz]);
```

[illegible]的多维数据。最后，将上面的程序代码保存为"blinnblob.m"文件。

Step 17 [illegible]

```
function slicebucky
% SLICEBUCKY - Make a bucky ball approximation for sliceomatic
  disp('Creating bucky ball approximation...');
  [b,v]=bucky;
  [illegible];
  v=blinnblob(centers,64,64,64);
  disp('Starting Sliceomatic');
  sliceomatic(v)
  daspect([1 1 1]);
  xlim([1 64]);
  ylim([1 64]);
  zlim([1 64]);
```

[illegible]

Step 18 [illegible]

```
>> slicebucky
系统会显示出下面的信息
Creating bucky ball approximation...
Starting Sliceomatic
Smoothing for IsoNormals...
Generating reduction volume...
```

Step 19 [illegible] 所示。

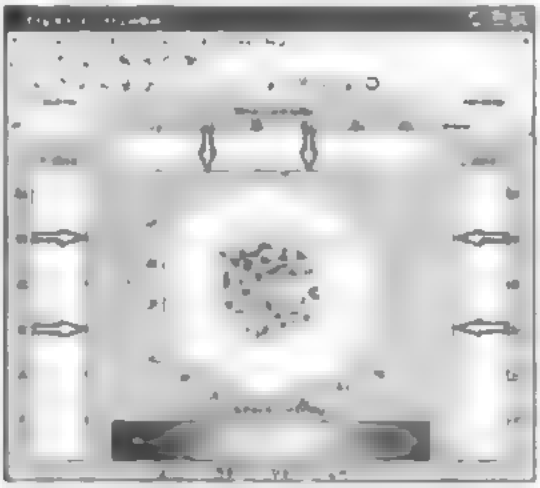


图 11.118 绘制 Slice 剖面

Step 20 [illegible] Isosurface 曲面，绘制的结果如图 11.119 所示。

[illegible]

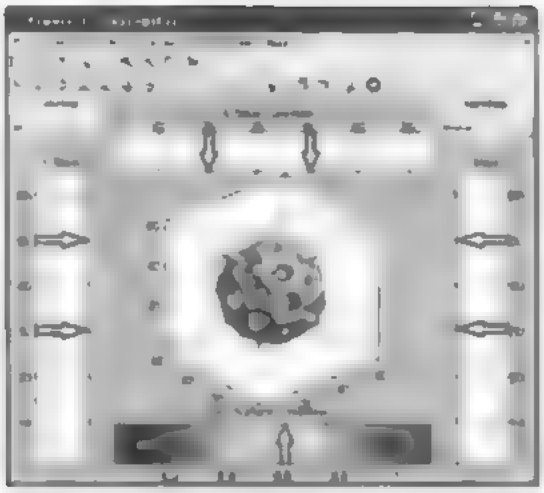


图 11.119 添加isosurface曲面

11.10 小结

[illegible]

第12章 文 件 I/O

本章包括

- 打开外部文件
- 关闭外部文件
- 读取二进制文件
- 写入二进制文件
- 读取文本文件
- 写入文本文件
- 处理图像对象

为了加强 MATLAB 的应用功能，实现 MATLAB 和其他格式的文件的相互交换是很重要的内容。MATLAB 系统具有直接对磁盘文件进行访问的功能，不仅可以进行高层次的程序设计，也可以对低层次的文件进行读写操作，这样就增加了 MATLAB 程序设计的灵活性和兼容性。在 MATLAB 中，提供许多有关文件输入和输出的函数，使用这些函数可以很方便地实现各种格式的读取工作，大多数函数都是基于 C 语言的文件 I/O 函数，因此比较容易上手，下面简要介绍。

处理文件名称

为了实现各种不同格式文件的读取工作，MATLAB 首先提供能够处理文件路径或者名称的函数，使用这些函数，用户可以对文件路径进行各种处理：分割路径名称、组合路径名称等，下面使用简单的例子来说明如何使用这些函数。

例 12.1 获取文件路径的各种信息，并对路径各部分进行处理。

step 1 编写处理路径的代码。选择编辑栏中的"File" ⇨ "New" ⇨ "M-file" 命令，打开 M 文件编辑器，然后在其中添加下面的程序代码：

```
function fileinfo(file)
%获取文件路径的各种信息
[pathstr,name,ext,versn] = fileparts(file);
%处理文件名称
if name=='fileinfo'
disp('This is the target file')
else
disp('This is not the target file')
end

%处理文件的后缀
if ext==strcat('.','xls')
disp('This is a excel file')
else
disp('This is not a excel file')
end
```

将上面的程序代码保存为"fileinfo.m"文件，在后面的程序代码中将需要调用该函数来处理文件的名称和后缀信息。

Step 2 [illegible]

```
>> file = '\home\smartchen\matlab\fileinfo.xls';
>> fileinfo(file)
This is not the target file
This is a excel file
>> file = '\home\smartchen\matlab\fileinfo.xls';
>> fileinfo(file)
This is the target file
This is a excel file
>> file = '\home\smartchen\matlab\fileinfo.txt';
>> fileinfo(file)
This is the target file
This is not a excel file
```

[illegible]函数来返回文件路径各部分的信息，其完整的调用格式为

```
[pathstr,name,ext,versn] = fileparts('filename')
```

[illegible]，pathstr 是文件路径，name 是文件名称，ext 是文件的后缀（包含后缀前面的点号），versn 返回的是文件版本。

[illegible]

例 12.2 利用路径各部分的内容创建完整的文件路径。

Step 1 [illegible]

```
>> file = '\home\smartchen\matlab\fileinfo.xls';
>> [pathstr,name,ext,versn] = fileparts(file);
>> filecontr=fullfile(pathstr,[name ext versn]);
```

Step 2 [illegible]

```
>> pathstr
pathstr =
[illegible]
>> name
name =
[illegible]
>> ext
ext =
.xls
>> filecontr
filecontr =
[illegible]
```

[illegible]函数，在 MATLAB 中，使用[illegible]命令根据各部分内容创建完整的文件路径，其完整的调用格式如下

```
f = fullfile('dir1','dir2',...,'filename')
```

[illegible]，[illegible]，文件名缺中如果不包含后缀，则创建的完整路径也不包含后缀。

[illegible]

12.2 打开和关闭文件

在对文件进行处理的所有工作当中，打开文件或者创建文件都是最基础的操作。在本小节中，将介绍如何在MATLAB中打开和关闭文件。根据操作系统的要求，在打开文件时需要使用或者创建一个磁盘文件的句柄。[illegible]操作系统发出打开文件的命令，使用完毕之后，也会[illegible]操作系统发出关闭文件的命令。

对于在MATLAB中编辑的文件，可以使用load、save等命令对该文件进行操作，其操作方法请参考第3章的内容。在本小节中，将主要介绍如何在MATLAB中读取低层次的数据文件方法。

12.2.1 打开文件

在本小节中，将主要使用简单的实例介绍如何在MATLAB中打开磁盘中的文件，然后详细介绍相应命令的使用方法。

例12.3 在MATLAB中使用fopen命令打开磁盘文件。

step 1 以读写的方式打开磁盘文件fgetl.m。在MATLAB的命令窗口中输入下面的代码

```
>> [fid,message] = fopen('fgetl.m','r+')
```

step 2 查看程序结果。在输入上面的代码后，得到的结果如下

```
fid =
     [illegible]

message =
No such file or directory;
```

由于在运行该命令的时候，命令直接在当前目录中寻找用户定义的文件路径为：MATLAB7.0work，而fgetl.m是系统内部的M文件，默认保存路径为MATLAB7.0toolbox/matlab/iofun。

step 3 查看M文件的源代码，方便后面步骤中打开的文件为系统文件。下面为生成文件的代码

```
function tline = fgetl(fid)
try
    [tline,lt] = fgets(fid);
    tline = tline(1:end-length(lt));
    if isempty(tline)
        tline = '';
```

```
    if nargin ==
        [illegible]
    end
    if isempty(fopen(fid))
        error('MATLAB:[illegible]','Invalid file identifier.')
    end
    [illegible]
end
```

[illegible]

step 4 [illegible] MATLAB 命令窗口，输入下面代码：

```
>> [fid,message] = fopen('fgetl.m','w')
```

step 5 查看程序结果。上面的程序代码得到的结果如下

```
fid =
     [illegible]
message =
     [illegible]
```

[illegible]

[illegible]

step 6 [illegible] MATLAB 命令窗口中输入下面代码：

```
>> [fid,message] = fopen('fgetl.m','r+')
```

step 7 查看程序代码的结果。上面的程序代码可以得到下面的结果

```
fid =
     [illegible]
message =
     [illegible]
```

[illegible]

上面给了基本章节"MATLAB中fopen命令的使用方法"，其对应的完整调用格式为：

- [fid,message] = fopen(filename, mode)
- [fid,message] = fopen(filename, mode, machineformat)

在上述的命令中，filename表示要打开文件的名称，mode表示打开文件方式，其取值类型如下。

- "r"：以只读方式打开文件。
- "w"：以只写方式打开文件，并覆盖原来的内容。
- "a"：增补文件，在文件尾部增加数据。
- "r+"：读写文件。
- "w+"：[illegible]
- "a+"：读取和增补文件。

[illegible]

在上述的命令中，fid是一个整数，一般称呼为文件标示。在MATLAB中，用户对文件的任何操作，都需要通过fid标数来传递，MATLAB会根据fid的数值来知道哪有已经打开的文件，然后对该文件的读、写和其他各种操作。如果给定的fid的数值是-1，则表示fopen不能打开对应的文件，可能是因为该文件本身不存在，用户没有读写的方式来打开，或者文件存在但是不在搜索路径上。

[illegible]

12.2.2 关闭文件

在前面小节中曾经介绍过，在对文件操作，如果完成了对应的读写工作，应该关闭文件，否则打开的文件数会增多，造成系统资源的浪费。在本小节中将以一个简单的实例来说明如何在MATLAB中关闭对应的文件。

例 12.4 在MATLAB中关闭对应的磁盘文件。

step 1 创建文件fgetl.m，然后删除该文件。在MATLAB的命令窗口中输入下面的代码：

```
>> [fid,message] = fopen('fgetl.m','w');
>> delete fgetl.m
```

step 2 查看运行代码的结果，下面的代码中[illegible]

```
Warning: File not found or permission denied.
```

[illegible]

Step 5 [illegible]

```
[illegible]
[illegible]
>> [fid,message] = fopen('foetl.m','r+');
```

Step 6 [illegible]

```
fid =
    -1
message =
No such file or directory
```

[illegible]

上述实例中，文件[illegible]在MATLAB中[illegible]文件。在MATLAB中，用户可以使用fclose的命令来关闭打开的文件，其具体的调用命令如下

```
status = fclose(fid)
status = fclose('all')
```

[illegible]，使用命令删除[illegible]打开的文件。如果使用该命令，需要检查结果status=0，表示关闭文件的操作成功；否则得到的结果status=-1。

[illegible]

12.3 处理二进制文件

在MATLAB中[illegible]，需要[illegible]文件或者XML文件[illegible]，这些文件[illegible]在MATLAB中[illegible]。本节中[illegible]，用户可以读取和写入二进制文件。

12.3.1 读取M文件

[illegible]的文件。本小节还是以具体的实例来说明如何使用该命令。

例 12.5 在MATLAB中读取ball.m文件的内容。

Step 1 [illegible]

M文件编辑器查看该文件的代码内容，得到结果如下：

```
%  Constants
conv=pi/180;
grav=-9.82;
vo=input('Enter the initial velocity:');
range=zeros(1,91);
% 计算最长的水平距离
for ii=1:91
     theta=ii-1;
     vxo=vo*cos(theta*conv);
     vyo=vo*sin(theta*conv);
     max_time=-2*vyo/grav;
     range(ii)=vxo*max_time;
end
% 读入水平距离的列表
fprintf('Range versus angle theta:\n');
for ii=1:5:91
    theta=ii-1;
    fprintf('%2d %8.4f\n',theta,range(ii));
end
% 计算最大的角度和水平距离
[maxrange index]=max(range);
maxangle=index 1;
fprintf('\n Max range is %8.4f at %2d degrees.\n',maxrange,maxangle);
% 绘制轨迹
for ii=5:10:80
     theta=ii;
     vxo=vo*cos(theta*conv);
     vyo=vo*sin(theta*conv);
     max_time=-2*vyo/grav;
     % 计算坐标数值x, y
     x=zeros(1,21);
     y=zeros(1,21);
     for jj=1:21
         time=(jj-1)*max_time/20;
         x(jj)=vxo*time;
         y(jj)=vyo*time+0.5*grav*time^2;
     end
     plot(x,y,'g');
     if ii==5
         hold on;
     end
end
% 添加标题和坐标轴名称
title('\bf Trajectory of Ball vs Initial Angle\theta');
xlabel('\bf\itx \rm\bf(meters)');
ylabel('\bf\ity \rm\bf(meters)');
axis([0 max(range)+5 0 -vo^2/2/grav]);
grid on;
% 绘制最长距离的轨迹曲线
vxo=vo*cos(maxangle*conv);
vyo=vo*sin(maxangle*conv);
max_time=-2*vyo/grav;
     % 计算坐标轴数值x, y
     x=zeros(1,21);
```

```
[illegible]
for jj=1:21
    [illegible]
    [illegible]
    y(jj)=vyo*time+0.5*grav*time^2;
end
plot(x,y,'r','Linewidth',2);
hold off
```

说明 [illegible]

step 2 读取[illegible]文件后，在MATLAB的命令窗口中输入下面的代码

```
>> [fid,message]=fopen('ball.m','r+');
>> data=fread(fid);
```

当用户读取上面的文件后，可以查看data变量的属性

```
>> whos data
  Name       Size                    Bytes  Class
  data     2522x1                    20176  double array

Grand total is 2522 elements using 20176 bytes
```

说明 [illegible]

step 3 查看读取的数据结果，在MATLAB的命令窗口中输入下面的代码

```
>> data
```

step 4 查看[illegible]的结果，在命令窗口中输入变量名称，得到的结果如下

```
data =
    [illegible]
   ............// 省略了部分数据
    [illegible]
```

说明 [illegible]

step 3 查看程序代码在命令窗口中输出的结果

```
>> disp(char(data'))
上面程序代码得到的结果如下
% Constants
[illegible]
[illegible]
[illegible]
range=zeros(1,91);
% Calculate maximum ranges
for ii=1:9;
    [illegible]
    vxo=vu*cos(theta*conv);
    [illegible]
    max_time=[illegible]*vy/[illegible]grav;
    range(ii)=vxo*max_time;
end
-------------------------------------------------------------//省略了部分代码
        [illegible]*max_time/[illegible];
        x(ii)=vxo*time;
        y(ii)=[illegible]*time-[illegible]*grav*time^2.
    end
    plot(x,y,'r','Linewidth',2);
    hold off
```

[illegible]

12.3.2 读取 TXT 文件

TXT 文件也是比较常见的二进制文件。在本小节中，将以一个简单的例子来介绍如何在 MATLAB 中读取 TXT 文件。

例 12.6 在 MATLAB 中读取 readtxt.txt 文件中的内容，并对读取的内容进行处理。

step 1 在本实例中，readtxt.txt 是文本文件，其中的内容包含下面的字符串：

```
This is a txt file
```

step 2 读取 readtxt.txt 文件的内容。在 MATLAB 的命令窗口中输入下面的代码：

```
>> fid = fopen('readtxt.txt', 'r');
>> data=fread(fid);
```

step 3 查看程序代码的结果。在命令窗口中输入变量名称，得到的结果如下：

```
>> data
data =
    84
   104
   105
```

```
   115
    32
    [illegible]
    [illegible]
    [illegible]
    [illegible]
.....................//省略了部分内容
    [illegible]
    [illegible]
    [illegible]
    [illegible]
    [illegible]
    [illegible]
    10
```

[illegible]

step 4 读取 readtxt.txt 文件的部分内容。在 MATLAB 的命令窗口中输入下面的代码

```
>> fid = fopen('readtxt.txt', 'r');
c = fread(fid, 5)
```

step 5 查看程序代码的结果。在命令窗口中输入变量名称，得到的结果如下

```
c =
    84
   104
   105
   115
    [illegible]
```

[illegible]

step 6 读取 readtxt.txt 文件的部分文本内容。在 MATLAB 的命令窗口中输入下面的代码

```
>> fid = fopen('readtxt.txt', 'r');
>> c = fread(fid, '*char');
>> sprintf(c)
```

step 7 查看程序代码的结果。输入代码后，按"Enter"键，得到的结果如下

```
ans =

This [illegible] txt file
```

step 8 处理 readtxt.txt 文件的部分文本内容。在 MATLAB 的命令窗口中输入下面的代码

```
>> fid = fopen('readtxt.txt', 'r');
>>c1 = fread(fid, 5, '*char');
```

```
>>c2 = fread(fid, 3, '*char');
>>c3 = fread(fid, 2, '*char');
>> c4 = fread(fid, 4, '*char');
>> c5 = fread(fid, 5, '*char');
>> sprintf('%c', c1, ' * ', c2, ' * ', c3,' * ',c4,' * ',c5)
```

step 9 查看程序代码的结果。输入代码后，按"Enter"键，得到的结果如下：

```
ans =

This  * is  * a  * txt  * file
```

上面的例子并不复杂，基本上演示了如何在MATLAB中读取二进制文件的方法。在MATLAB中，读取二进制文件的命令是fread，其调用格式如下：

```
A = fread(fid, count, precision)
```

在上面的命令中，参数fid表示使用fopen命令打开的文件标识，参数count表示读取二进制文件的大小，可以选取下面三个参数。

- n：读取前面n个整数，并写入一个向量中。
- inf：读取文件，直到文件结尾处。
- [m,n]：读取数据到m*n的矩阵中，按照列排列，n可以是inf，但是m不能是inf。

最后一个参数precision用来控制二进制数据转换成为MATLAB矩阵时的精度，具体的精度参数请用户查看相应的帮助文件。

写入二进制文件

在MATLAB中，如果用户希望按照指定的二进制文件格式将矩阵的元素读入文件中，可以使用fwrite命令来完成这样的任务。和前面小节类似，在本小节中将使用简单的例子来说明如何使用该命令。

例 12.7 在MATLAB中使用fwrite命令来写入二进制文件。

step 1 在MATLAB的命令窗口中输入下面的程序代码：

```
>>fid = fopen('magic5.txt','wb');
>>fwrite(fid,magic(5),'int32');
>>fid=fopen('magic5.bin','r');
>>data=fread(fid,[ 5,5] ,'int32');
>>A=data';
```

step 2 查看程序代码的结果。在命令窗口中输入变量名称，得到的结果如下：

```
A =
    17    23     4    10    11
    24     5     6    12    18
     1     7    13    19    25
     8    14    20    21     2
    15    16    22     3     9
>> magic(5)
ans =
    17    24     1     8    15
```

[illegible]

[illegible]

12.4 处理文本文件

[illegible]

在本小节中，将通过实例来详细介绍。

12.4.1 读取文本文件

[illegible]

实例来说明这些函数的使用方法。

例 12.8 [illegible]

step 1 [illegible]

数据如下

```
[illegible], [illegible], 06, 08, 10, 12
03, 06, 09, 12, 15, 18
05, 10, 15, 20, 25, 30
07, 14, 21, 28, 35, 42
11, 22, 33, 44, 55, 66
```

step 2 [illegible]

```
>> m1 = csvread('txtlist.dat');
>> m2 = csvread('txtlist.dat', 2, 0);
[illegible] 0, [2,0,3,3]);
[illegible]
```

step 3 [illegible]

```
[illegible]
[illegible]
```

```
     5    10    15    20
     7    14    21    28
nd =
     2     4     6     8    10    12
     3     6     9    12    15    18
     5    10    15    20    25    30
     7    14    21    28    35    42
    11    22    33    44    55    66
```

[illegible]

例12.9 在MATLAB中使用textread命令来读取文本文件。

step 1 [illegible]

```
Sally    Level1 12.34 45 Yes
```

step 2 [illegible]

```
>>[ names, types, x, y, answer] = textread('txtlist2.txt', . .
'%s %s %f %d %s', 1);
```

step 3 [illegible]

```
names =
    [illegible]
types =
    [illegible]
x =
    [illegible]
y =
    [illegible]
answer =
    [illegible]
```

[illegible]

例12.10 在MATLAB中使用txtscan命令来读取文本文件。

step 1 [illegible]

```
Sally  Level1 12.34 45 1.23e10 inf NaN Yes
Joe    Level2 23.54 60 9e19 -inf 0.001 No
Bill   Level3 34.90 12 2e5 10 100 No
```

step 2 [illegible]

```
>> fid = fopen('txtscan.dat');
```

```
>> C1 = txtscan(fid, '%s %s %F32 %dB %u %f %f %s');
>> fclose(fid);
```

step 3 [illegible]

```
>> whos C1
  Name      Size                    Bytes  Class
  C1        1x8                      1169  cell array

Grand total is 69 elements using 1169 bytes
```

[illegible]

step 4 [illegible]

```
[illegible]
[illegible]
end
```

step 5 [illegible]

```
'Sally'    'Joe'    'Bill'
   [illegible]
   12.3400   23.5400   34.9000
   45   60   12
  4294967295  4294967295      200000
   Inf   -Inf    10
      NaN    0.0010  100.0000
   [illegible]
```

[illegible]

```
[illegible]
C1{2} = {'Level1'; 'Level2'; 'Level3'}     class cell
C1{3} = [12.34; 23.54; 34.90]              class single
[illegible]
C1{5} = [1.23e10; 9e19; 2e5]               class uint32
C1{6} = [Inf; -Inf; 10]                    class double
C1{7} = [NaN; 0.001; 100]                  class double
C1{8} = {'Yes'; 'No'; 'No'}                class cell
```

step 6 [illegible]

```
>> fid = fopen('txtscan.dat');
>>C2 = txtscan(fid, '%7c %6s %*2 %d8 %u %f %f %s');
[illegible]
```

step 7 [illegible]

```
>> whos C2
  Name      Size                    Bytes  Class
  C2        1x7                       935  cell array
```

```
Grand total is 71 elements using 935 bytes
```

step 8 查看C2数组的结果。在命令窗口中输入下面的程序代码

```
[illegible]
disp(C2{ 5} ');
end
```

step 9 查看程序的结果。输入[illegible]按"Enter"键，[illegible]

```
'[illegible]'    '[illegible]'    '[illegible]'
    '[illegible]'    '[illegible]'    '[illegible]'
   45    60    12
  4294967295  4294967295      200000
   Inf   -Inf     10
      NaN     0.0010   100.0000
    'Yes'    'No'    'No'
```

同样，将上面的结果整理如下

```
C2{ 1} = { 'Sally  '; 'Joe    '; 'Bill   '}    class char
C2{ 2} = { 'Level1'; 'Level2'; 'Level3'}       class cell
C2{ 3} = { 45; 60; 12}                         class int8
C2{ 4} = { 1.23e10; 9e19; 2e5}                 class uint32
C2{ 5} = { Inf; -Inf; 10}                      class double
[illegible]
[illegible]
```

说明 [illegible]

step 10 创建读取原始文件的第一个数据，[illegible]，并输入下面的程序代码

```
>> fid = fopen('txtscan.dat');
names = textscan(fid, '%s%[^\n]');
[illegible]
```

step 11 查看names的属性，在命令窗口中输入[illegible]

```
>> whos names
  Name        Size                    Bytes  Class
  names       1x1                       264  cell array

Grand total is 16 elements using 264 bytes
```

step 12 查看程序中的字符串，在命令窗口中输入[illegible]

```
>> B= names{1}'
B =
'[illegible]'    '[illegible]'    '[illegible]'
```

step 13 删除[illegible]，在命令窗口中输入[illegible]

```
>> fid = fopen('textscan.dat');
>>C3 = textscan(fid, '%s %d%d%u %f %f %d');
[illegible]
>> whos C3
```

Step 14 查看程序代码的结果。输入代码后，按"Enter"键，程序的结果如下

```
  Name      Size                    Bytes  Class
  C3        1x8                       956  cell array

Grand total is 51 elements using 956 bytes
```

Step 15 [illegible]

```
>> for i=1:8
[illegible]
```

Step 16 [illegible]

```
[illegible]
    1      2      3
  12.3400   23.5400   34.[illegible]
  45    60    12
[illegible]
  Inf   -Inf     10
     NaN    0.0010   100.0[illegible]
  'Yes'    'No'    'No'
```

说明 [illegible]

最后，将上面的结果整理如下

```
[illegible]
C3{2}  = [1; 2; 3]                    class uint8
[illegible]
C3{4}  = [45; 60; 12]                 class int8
C3{5}  = [1.2e10; 9e19; 4e5]          class uint32
C3{6}  = [Inf; -Inf; 10]              class double
C3{7}  = [NaN; 0.001; 100]            class double
[illegible]
```

根据上面的例子可以看出，在 MATLAB 中读取文本文件的命令如下。

- N = [illegible]，[illegible] row 应需要读取的行，col 则是需要读取的数据列。
- [illegible] 表示用户自行指定的分隔符。
- [illegible] format 则表示读取文件的变量格式，N 表示读取数据的循环次数。

12.4.2 使用csvwrite命令读入文本文件

[illegible]句在MATLAB中写入文本文件。

例 12.11 使用csvwrite命令向文本文件写入MATLAB的数据

Step 1 [illegible]

```
m = [3 6 9 12 15; 5 10 15 20 25; 7 14 21 28 35; 11 22 33 44 55];
[illegible]
[illegible]
```

Step 2 [illegible]

```
3,6,9,12,15
[illegible]
7,14,21,28,35
[illegible]
```

Step 3 [illegible]的代码

```
[illegible]
type csvlist.dat
```

Step 4 [illegible]

```
,,3,6,9,12,15
[illegible]
[illegible]
[illegible]
```

Step 5 [illegible]加2个数据[illegible]。在命令窗口输入下面的代码

```
[illegible]
type csvlist.dat
```

Step 6 [illegible]

```
[illegible]
[illegible]
,,,,3,6,9,12,15
[illegible]
[illegible]
[illegible]
```

[illegible]

使用 dlmwrite 命令读入文本文件

在MATLAB中，还可以使用dlmwirte命令来读入文本文件，下面使用具体的实例来说明如何使用该命令。

例 12.12 使用dlmwrite命令向文本文件写入MATLAB的数据。

step 1 将数据写入myfile文本文件中。在MATLAB的命令窗口中输入下面的程序代码：

```
>> m=rand(6);
dlmwrite('myfile.txt', m, 'delimiter', '\t', 'precision', 5)
type myfile.txt
```

step 2 查看程序代码的结果。输入代码后，按"Enter"键，得到的结果如下：

```
0.84622	0.68128	0.30462	0.15087	0.49655	0.34197
0.52515	0.37948	0.18965	0.6979	0.89977	0.28973
0.20265	0.8318	0.19343	0.37837	0.82163	0.34119
0.67214	0.50281	0.68222	0.86001	0.64491	0.53408
0.83812	0.70947	0.30276	0.85366	0.81797	0.72711
0.01964	0.42889	0.54167	0.59356	0.66023	0.30929
```

step 3 修改数据精度，然后将数据写入myfile文本文件中。输入下面的程序代码：

```
>> m=rand(6);
dlmwrite('myfile.txt', m, 'delimiter', '\t', 'precision', 3)
type myfile.txt
```

step 4 查看程序代码的结果。输入代码后，按"Enter"键，得到的结果如下：

```
0.838	0.695	0.173	0.137	0.284	0.516
0.568	0.621	0.98	0.0118	0.469	0.334
0.37	0.795	0.271	0.894	0.0648	0.433
0.703	0.957	0.252	0.199	0.988	0.226
0.547	0.523	0.876	0.299	0.583	0.58
0.445	0.88	0.737	0.661	0.423	0.76
```

step 5 向myfile文本文件中写入多行数据。输入下面的程序代码：

```
>> M = ones(5);
dlmwrite('myfile.txt', [M*5 M/5], ' ')
dlmwrite('myfile.txt', eye(4),'-append', ...
   'roffset', 1, 'delimiter', ' ')
type myfile.txt
```

step 6 查看程序代码的结果。输入代码后，按"Enter"键，得到的结果如下：

```
5 5 5 5 5 0.2 0.2 0.2 0.2 0.2
5 5 5 5 5 0.2 0.2 0.2 0.2 0.2
5 5 5 5 5 0.2 0.2 0.2 0.2 0.2
5 5 5 5 5 0.2 0.2 0.2 0.2 0.2
5 5 5 5 5 0.2 0.2 0.2 0.2 0.2

1 0 0 0
0 1 0 0
```

[illegible]

从上述的[illegible]，在MATLAB中，[illegible]的调用格式如下。

- [illegible]，filename表示[illegible]的文件名称，M是对应的数据矩阵。row和col表示在[illegible]中的起始行和列。
- dlmwrite('filename', M, '-append', attribute-value list)：在该命令中，filename表示数据写入的文件名称，M是对应的数据矩阵，'-append'表示保留原来的[illegible]。最后的属性表[illegible]，用户可以带[illegible]的参数属性，具体属性可参阅帮助文件。

12.5 处理图像

在MATLAB中，图像一直是十分重要的组成部分，MATLAB可以在图像处理方面为您提供帮助，并可为各种复杂的工作。MATLAB还专门提供Image Acquisition、Image Processing等工具箱，来完成各种复杂的图像处理工作。本节将以一个简单的实例来说明如何在MATLAB中读取和编辑图像对象。

例12.13 在MATLAB中读入某图像文件，并进行适当的处理。

step 1 读入图像文件。输入下面的程序代码：

```
>> RGB = imread('gantryorane.png');
imshow(RGB);
```

step 2 查看读入结果。输入代码后，按"Enter"键，[illegible]

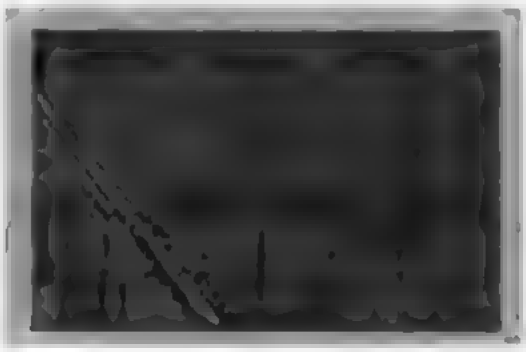


图12.1 读入图像文件

step 3 查看图像信息数组。在MATLAB命令窗口输入下面的程序代码：

```
>> whos
  Name        Size                    Bytes  Class
  RGB       264400x3                 316800  uint8 array

Grand total is 316800 elements using 316800 bytes
```

在MATLAB中，[illegible]，并且能从命令窗口[illegible]。在[illegible]中，[illegible]，[illegible]，[illegible]，[illegible]图像文件的数据大小。

Step 4 [illegible]

```
[illegible]
[illegible]
  [illegible]
line([ 300 [illegible]
line([ [illegible]

[illegible]
[illegible],'[illegible] between these beams',[illegible],[illegible],
    [illegible],[illegible]);
```

Step 5 单击制表对象，输入代码后，按"Enter"键，保存的图像效果如图 12.2 所示。

图 12.2 添加说明文字后的图像

Step 6 选择需要处理的图像，在命令窗口中输入下面的代码：

```
% you can obtain the coordinates of the rectangular region using
% pixel information displayed by imview
start_row = 34;
start_col = 200;

[illegible] = RGB(start_row:[illegible], start_col:400, :);

imshow(cropRGB)

% Store (X,Y) offsets for later use; subtract 1 so that each offset
%
[illegible]rrespond to the last pixel before the region of interest
[illegible] = start_col-1;
[illegible] = start_row-1;
```

Step 7 单击制表对象，输入代码后，按"Enter"键，保存的图像效果如图 12.3 所示。

说明

在[illegible]

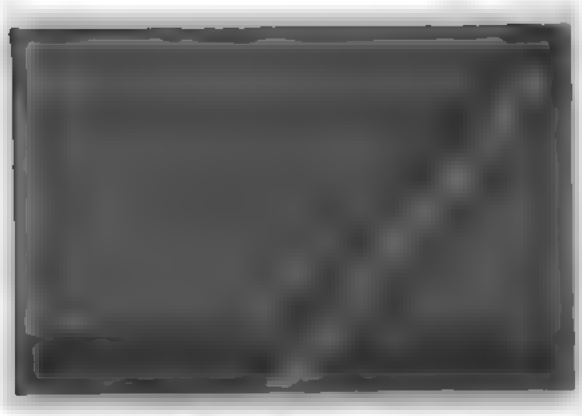


图 12.3 [illegible]

step 8 [illegible]

```
[illegible]
[illegible]
[illegible]
BW = [illegible]
[illegible] = BW; [illegible] interest must be white
[illegible]
```

step 9 [illegible] 输入[illegible]，按 "Enter" 键，[illegible]

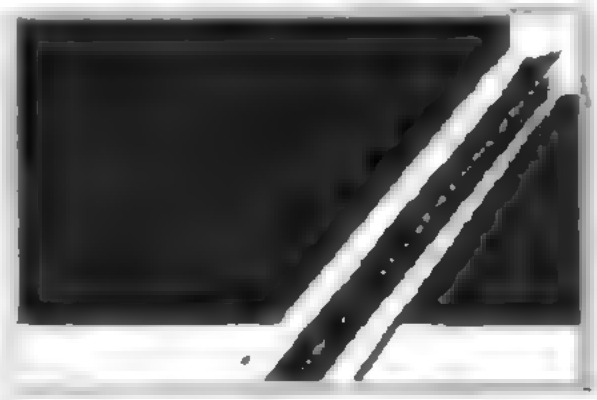


图 12.4 [illegible]

step 10 计算完成之后，[illegible]，在命令窗口中输入下面的程序代码

```
[illegible] = size(BW);
[illegible]
[illegible] = 4;
row1 = [illegible];
[illegible]
row2 = [illegible];
[illegible] = [illegible];
[illegible] = bwtraceboundary(BW, [row1, col1], 'N', 8, Inf);
[illegible] kwise, in order to trace downward.
boundary2 = bwtraceboundary(BW, [row2, col2], 'E', 8, 90,'counterclockwise');
imshow(BW); hold on;
[illegible] Draw in the original image
[illegible]boundary1(:,1),'g','LineWidth',[illegible]
[illegible] offsetY+boundary2(:,1),'g','LineWidth',[illegible]
```

Step 11 [illegible] 输入代码，按"Enter"键，[illegible]

图 12.5 向图像文件中添加边界线

Step 12 计算两个边界线之间的夹角。在命令窗口中输入下面的程序代码

```
[illegible]
ab2 = polyfit(boundary2(:,2), boundary2(:,1), 1);
[illegible] vect1 = [1 ab1(1)]; % create a vector based on the line equation
vect2 = [1 ab2(1)];
dp = dot(vect1, vect[illegible]
[illegible]
length1 = sqrt(sum(vect1.^2));
length2 = sqrt(sum(vect2.^2));
% obtain the larger angle of intersection in degrees
[illegible] 90-acos(dp/(length1*length2))*180/pi;
```

[illegible]

Step 13 计算两个边界线的交点。在命令窗口中输入下面的程序代码

```
intersection = [1 ,-ab1(1); 1, -ab2(1)] \ [ab1(2); ab2(2)];
% apply offsets in order to compute the location in the original,
% i.e. not cropped, image.
intersection = intersection + [offsetY; offsetX]
```

Step 14 显示计算结果。在命令窗口中输入下面的程序代码

```
[illegible] = intersection(2);
[illegible] = intersection(1);
% [illegible] of "%" at the [illegible]
[illegible]
[illegible] sprintf('%2.3f',angle),'(\circ)'],...
[illegible]
interString = sprintf('( %.1f,%2.1f)', inter_x, inter_y);
text(inter_x-10, inter_y+20, interString,...
    'Color','[illegible]','[illegible]','[illegible]');
```

Step 15 在命令窗口中输入[illegible]，按"Enter"键，保存[illegible]

图 12.6 显示计算结果

12.6 小结

在本章中，主要向读者介绍了打开外部文件、关闭外部文件、读取二进制文件、写入二进制文件、读取文本文件、写入文本文件和处理图像对象的内容，通过这些基础内容读者可以了解MATLAB中如何进行文件输入和输出。在后面的章节中，将介绍如何使用MATLAB的编译器。

第13章 MATLAB编译器

- 安装编译器
- 配置编译器
- 编译命令
- 编译过程
- 创建独立的应用程序

[illegible]

13.1 编译器概述

[illegible]

13.1.1 编译器的功能

[illegible]

- **独立运行的程序**：[illegible]同时运行，它们可以在没有安装MATLAB7.0的机器上运行。
- **C和C++共享库（在Microsoft Windows操作系统中为动态连接库DLL）**：[illegible]在没有安装MATLAB7.0的用户机器上运行。
- **Excel插件**：需要安装MATLAB7.0Builder for Excel。
- **COM对象**：需要安装MATLAB7.0Builder for COM。

[illegible]程序。

[illegible]

13.1.2 Complier 4.0的性能改进

[illegible]

[illegible]

- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]Microsoft Windows和Linux操作系统中使用Comp[illegible] 4.0。

提示

关于Compiler[illegible]的具体使用方法[illegible]，或[illegible]请查看MATLAB[illegible]Compiler[illegible]。

13.2 编译器的安装和配置

在实现MATLAB编译器多种功能之前，需要首先安装MATLAB的编译器以及其他标准的C/C++编译器。本节将主要介绍编译器的安装和配置。

13.2.1 前提准备

当用户首次使用标准编译器时，MATLAB自动对其进行[illegible]。如果用户对标准编译器有特殊的要求，可以自行手动设置编译器的配置。

step 1 安装下面任何一种可以和MATLAB兼容的ANSI C/C++编译器。

- LCC：MATLAB自带的编译器，只能编译C代码，不能编译C++。
- Borland C++：可以接受的版本为5.3、5.4、5.5和5.6。
- Microsoft Visual C/C++：[illegible]

说明

若要选择[illegible]，[illegible]MATLAB[illegible]，[illegible]C代码，[illegible]C++代码，[illegible]。[illegible]，而不要[illegible]。建议使用完全安装。

step 2 安装MATLAB7.0的Compiler 4.0。

在默认情况下，MATLAB7.0中的Compiler的安装[illegible]在MATLAB7.0的安装过程中，[illegible]选择的是Typical（典型）安装模式的，MATLAB7.0的Compiler会自动安装。当用户选择的是自定义安装模式的时候，Compiler选项处于默认状态，[illegible]，则[illegible]安装Compiler 4.0组件，如图13.1所示。

图 13.1 安装 Compiler 4.0

13.2.2 配置编译器

前面做好了安装编译器的准备工作，本小节中将要对编译器进行配置。由于编译器的配置与用户所使用的系统属性有关，本章只介绍了如何安装其他编译器的情况下，如何手动配置编译器。对于安装了其他类型编译器的读者，其设置工作和本节一样。

例 13.1 在 MATLAB 和 C 编译器应用程序 MEX 进行配置。

step 1 启动配置。在 MATLAB 的命令窗口中输入 mex -setup，系统出现下面的提示

```
Please choose your compiler for building external interface (MEX) files:
Would you like mex to locate installed compilers [y] /n?
```

step 2 启动 MATLAB 的自动寻找系统，出现下面的提示内容。如果用户选择 y，mex 将会自动搜索外部编译器的类型、版本以及所在的路径。MATLAB 会给出搜索结果，也就是系统所安装的所有外部编译器，并提示用户输入对应的数字

```
Select a compiler:
[1] Lcc C version 2.4 in D:\SOFTWARE\MATLAB7.0\sys\lcc
[0] None
Compiler:
```

step 3 确定选择的编译器类型。由于在笔者的系统中，没有安装其他的编译器，所以只有两个选项。当用户选择默认的编译器后，系统会要求用户确定选择

```
Please verify your choices:
Compiler: Lcc C 2.4
Location: D:\SOFTWARE\MATLAB7.0\sys\lcc
Are these correct?([y]/n):
```

step 4 结束配置。如果上面的定位信息没有错误，可以键入 y，结束编译器的配置工作，MATLAB 会显示结束信息

```
Try to update options in D:\Documents and Settings\Administrator\Application
Data\MathWorks\MATLAB\R14\mexopts.bat
From template:                    D:\SOFTWARE\MATLAB7.
0\BIN\WIN32\mexopts\lccopts.bat

Done . . .
```

[illegible]

例 13.2 验证编译器的配置工作，编译MEX文件。

step 1 [illegible]

```
#include "mex.h"
/* Input Arguments */
#define  T_IN prhs[0]
#define  Y_IN prhs[1]
/* Output Arguments */
#define  YP_OUT     plhs[0]
#if !defined(MAX)
#define  MAX(A, B)  ((A) > (B) ? (A) : (B))
#endif
#if !defined(MIN)
#define  MIN(A, B)  ((A) < (B) ? (A) : (B))
#endif
#define PI 3.14159265
static   double     mu = 1/82.45;
static   double     mus = 1 - 1/82.45;
static void yprime(
            double  yp[ ],
            double  t,
            double  y[ ]
            )

    [illegible]
    r1 = sqrt((y[0]+mu)*(y[0]+mu) + y[2]*y[2]);
    r2 = sqrt((y[0]-mus)*(y[0]-mus) + y[2]*y[2]);
    /* Print warning if dividing by zero. */
    if (r1 == 0.0 || r2 == 0.0 ){
    mexWarnMsgTxt("Division by zero!\n");
    }
    yp[0] [illegible]
     yp[1]  = 2*y[3]+y[0]-mus* [illegible] (r2*r2*r2);
    [illegible]
    yp[3]  = -2*y[1]  + y[2]  - mus*y[2]/(r1*r1*r1) - mu*y[2]/(r2*r2*r2);
    [illegible]

void mexFunction( int nlhs, mxArray *plhs[],
            int nrhs, const mxArray*prhs[] )

    double *yp;
    double *t,*y;
    unsigned int m,n;
    /* Check for proper number of arguments */
    if (nrhs [illegible])
```

```
[illegible] "[illegible] requires [illegible]",
} else if (nlhs > 1) {
mexErrMsgTxt("[illegible] many output arguments.");
}
/* Check the dimensions of Y.  Y can be 4 X 1 or 1 X 4. */
[illegible]
m = mxGetM(Y_IN);
if (!mxIsDouble(Y_IN) || mxIsComplex(Y_IN) ||
(MAX(m,n) != 4) || (MIN(m,n) != 1)) {
mexErrMsgTxt("YPRIME requires that Y be a 4 x 1 vector.");
}
/* Create a matrix for the return argument */
YP_OUT = mxCreateDoubleMatrix(m, n, mxREAL);
 /* Assign pointers to the various parameters */
yp = mxGetPr(YP_OUT);
t = mxGetPr(T_IN);
y = mxGetPr(Y_IN);
/* Do the actual computations in a subroutine */
[illegible]
return;
```

Step 2 [illegible]

```
function yp = yprime(t,y)
mu = 1/82.45;
mus = 1-mu;
r1 = norm([y(1)+mu, y(3)]);    % Distance to the earth
r2 = norm([y(1)-mus, y(3)]);   % Distance to the moon
yp(1) = y(2);
yp(2) = 2*y(4) + y(1) - mus*(y(1)+mu)/r1^3 - mu*(y(1)-mus)/r2^3;
yp(3) = y(4);
yp(4) = -2*y(2) + y(3) - mus*y(3)/r1^3 - mu*y(3)/r2^3;
```

[illegible]

Step 3 [illegible]

```
>> yprime(1,1:4)
```

Step 4 [illegible]

```
ans =
    2.0000    0.9685    4.0000   -1.0947
```

[illegible]

step 5 [illegible]

[illegible]编译执行文件保存在原来MEX文件相同路径中

例13.3 [illegible]

step 1 [illegible]

```
[illegible]
```

[illegible]

step 2 [illegible]

step 3 [illegible]

图13.2 在DOS条件下执行文件

说明 [illegible]

[illegible]具体的过程如下

```
[illegible]
Please choose your compiler for building standalone MATLAB
applications:
Would you like mbuild to locate installed compilers [y]/n? n
[illegible]
[1] Borland C++Builder version 6
[2] Borland C++Builder version 5.
[3] Borland C++Builder version 4
[4] Borland C++Builder version 3
[5] Borland C/C++ version 5.02
[6] Borland C/C++ version 5.0
[7] Borland C/C++ (free command line tools) version 5.5
[8] Lcc C version 2.4
[9] Microsoft Visual C/C++ version 7.1
[10] Microsoft Visual C/C++ version 7.0
[11] Microsoft Visual C/C++ version 6.0
[0] None
Compiler: 11
Your machine has a Microsoft Visual C/C++ compiler located at
[illegible] want to use this
compiler [y]/n? y
Please verify your choices:
Compiler: Microsoft Visual C/C++ 6.0
Location: D:\Applications\Microsoft Visual Studio
Are these correct?([y]/n): y
Try to update options file:
[illegible]
Data\MathWorks\MATLAB\R14\compopts.bat
[illegible]
\\sys\MATLAB\BIN\WIN32\mbuildopts\msvc60compp.bat
Done . . . .
Updated ...
```

提示 [illegible]

13.3 编译过程

[illegible]编译过程。

13.3.1 安装 MCR

[illegible]

例 13.4 在 Windows 操作系统中安装 MCR 组件。

step 1 [illegible]

step 2 [illegible]

图 13.3 安装 MCR 的界面

step 3 [illegible]

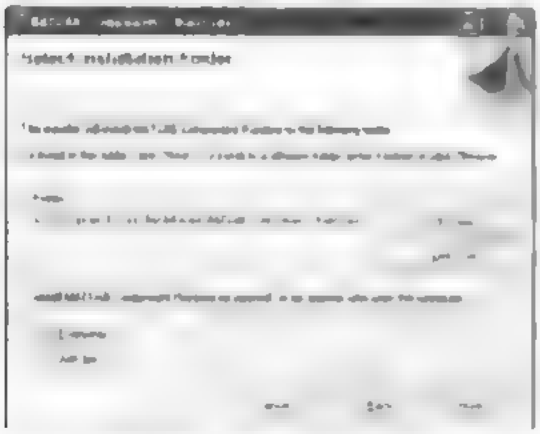


图 13.4 选择安装路径

[illegible]

step 4 [illegible]

安装，如图 13.5 所示。

图 13.5　MCR 安装过程界面

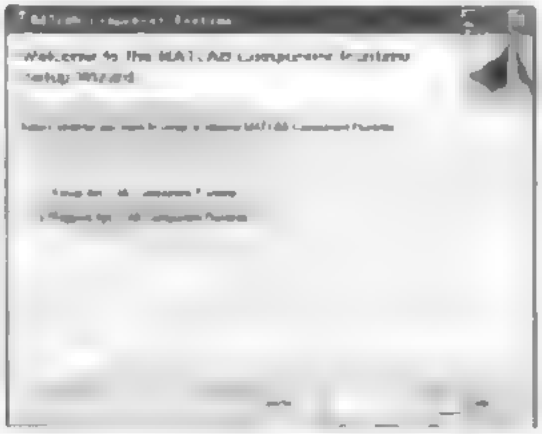


图 13.6　修复或者卸载 MCR 的界面

13.3.2 代码的编译过程

所有基于 MATLAB7.0的 M 文件或者 MEX 文件等。

自动运行对应的程序过程。

件的过程。

例 13.5　演示 MATLAB 中编译器编译文件的过程。

Step 1　绘制编译过程的流程图，如图 13.7 所示。

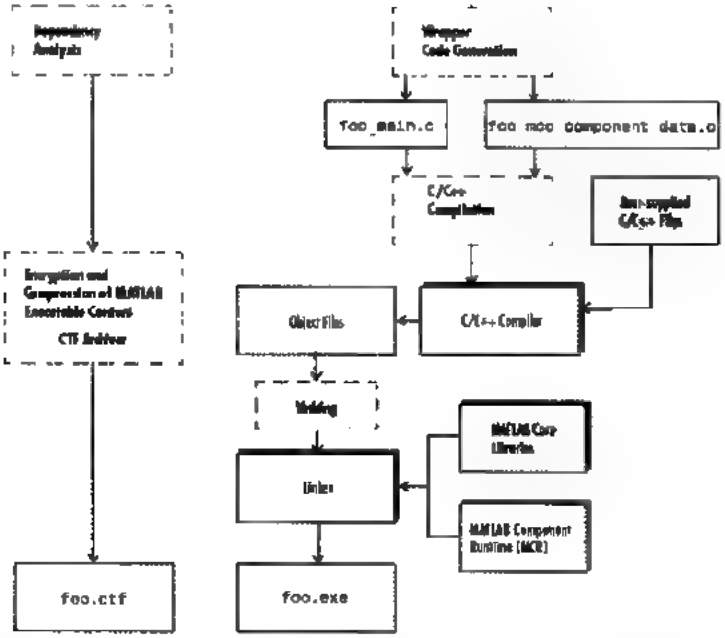


图 13.7 编译文件的流程图

step 2 解读编译过程的流程图。在本步骤中，将结合编译原理来说明编译文件的过程。

- **依赖性分析（Dependency analysis）**：分析判断输入的M文件、MEX文件以及P文件所依赖的函数之间的关系，这些函数包括了输入文件所调用的M文件以及MATLAB提供的函数。
- **代码生成（Code generation）**：生成所有用来产生目标组件的代码，包括从命令行获得的M函数相关的C和C++接口代码。对于共享库和组件来说，这些代码包括所有的接口函数代码。
- **存档生成（Archive creation）**：在依赖性分析中生成的MATLAB7.0可执行文件列表用来生成CTF存档文件，其中包括程序运行时所需组件的数据，存档在加密后被压缩在一个单独的文件中，同时路径信息也被保存。
- **编译（Compilation）**：编译生成的C和C++文件，得到目标代码。对于那些含有用户提供C或者C++代码的目标，这些代码同样也会被编译。
- **链接（Linking）**：将生成的目标文件以及相关的MATLAB的共享库链接起来，生成最终的组件。

编译命令

在MATLAB中提供mcc命令来进行编译工作。本节将主要介绍mcc命令的使用方法和注意事项；关于比较复杂的编译文件、方法以及过程将在后面详细介绍。

编译命令的格式和选项

无论希望生成一种或者多种应用程序，或者希望创建一个C共享库以及软件组件，只要源文件是M文件，都可以使用编译命令mcc来实现，可以在MATLAB环境以及DOS或者UNIX环境中使用该命令。

可以使用一种或者多种MATLAB编译器参数给mcc命令，大部分参数的名字由一个字母组成，用户可以独立地使用两个参数，如下面的示例代码：

```
mcc -m -g myfun
```

[illegible]

[illegible]

[illegible]

13.4.2 处理脚本文件

[illegible]

例13.6 [illegible]

Step 1 [illegible]

```
[illegible]
% 常数
[illegible]
grav=-9.82;
vo=45;
[illegible]
% 计算最长的水平距离
for ii=1:9;
    [illegible]
    [illegible]
    vyo=vo*sin(theta*conv);
    [illegible]
    range(ii)=vxo*max_time;
end
% 写入水平距离的列表
fprintf('Range versus angle theta:\n');
for ii=1:5:91
    [illegible]
    [illegible]
end
% 计算最长的角度和水平距离
[maxrange index]=max(range);
[illegible]
[illegible]
```

[illegible]

step 2 运行脚本文件 Compli_ball。在 MATLAB 的命令窗口中输入 "Compli_ball"，按回车键运行脚本文件，运行结果

```
>> Compli_ball
Range versus angle theta:
 0   0.0000
 5  35.8083
10  70.5286
15 103.1059
20 132.5504
25 157.9674
30 178.5847
35 193.7757
40 203.0790
45 206.2118
50 203.0790
55 193.7757
60 178.5847
65 157.9674
70 132.5504
75 103.1059
80  70.5286
85  35.8083
90   0.0000

 Max range is 206.2118 at 45 degrees.
```

[illegible]

step 3 添加函数声明行。在原来的脚本文件的开头处添加下面的程序代码

```
function Compli_ball
```

添加了上面的程序代码后，保存原来的脚本文件。

step 4 编译脚本文件。在 MATLAB 的命令窗口中输入下面的程序代码

```
>> mcc -m Compli_ball.m
```

step 5 运行可执行文件。运行[illegible]窗口，在上面步骤保存的路径中，运行编译产生的 Compli_ball.exe 文件，得到的结果如图 13.8 所示。

[illegible]

图 13.8　在 DOS 窗口中运行编译文件结果

13.5　创建独立运行的程序

[illegible]

13.5.1　编译 M 文件

[illegible]

例 13.7　编译 M 文件 timing.m，[illegible]

step 1　查看 M 文件 timing.m 的内容，其内容如下：

[illegible]

```
[illegible]
ave2=(toc)/maxcount;
%使用向量化结构
maxcount=100;
tic;
[illegible]
clear square
[illegible]
end
ave3=(toc)/maxcount;
%显示计算结果
fprintf('Loop/uninitialized array= %9.5f\n',ave1);
[illegible]
[illegible]
```

step 2 将 M 文件 timings.m 保存为 [illegible]，在 MATLAB 命令窗口中输入下面语句

```
>> timings
Loop/uninitialized array=   0.34100
Loop/initialized array=   0.00020
vectorized=   0.00010
```

[illegible]

step 3 编译 M 文件。在 MATLAB 命令窗口中输入下面语句

```
>> mcc -mv timings.m
```

step 4 出现系统提示信息，输入代码后，按 "Enter" 键，编译结束如下

```
Compiler version: 4.0 (R14)
Parsing file "d:\software\matlab7.0\work\timings.m"
    (Referenced from: "Compiler Command Line").
Parsing file "d:\software\matlab7.0\toolbox\compiler\deploy\matlabrc.m"
    (Referenced from: "Compiler Command Line").
Parsing file "d:\software\matlab7.0\toolbox\compiler\deploy\utils.m"
    (Referenced from: "Compiler Command Line").
Parsing file "d:\software\matlab7.0\toolbox\compiler\deploy\utils.m"
    (Referenced from: "d:\software\matlab7.0\toolbox\compiler\deploy
\matlabrc.m").
Parsing file "d:\software\matlab7.0\toolbox\matlab\strfun\strcmpi.m"
    (Referenced from: "d:\software\matlab7.0\toolbox\compiler\deploy
\matlabrc.m").
Parsing file "d:\software\matlab7.0\toolbox\matlab\private\[illegible]"
    (Referenced from: "d:\software\matlab7.0\toolbox\compiler\deploy
\matlabrc.m").
Parsing file "d:\software\matlab7.0\toolbox\matlab\[illegible]"
    (Referenced from: "d:\software\matlab7.0\toolbox\compiler\deploy
\matlabrc.m").
Warning: No output files [illegible] d:\software\MATLAB
[illegible]
Generating file "timings_main.c".
```

```
Depfun main loop, iteration 1
Processing D:\SoftWare\MATLAB7.0\toolbox\matlab\mcc.enc
1 items added
Processing dependencies...
0 items added
Depfun main loop, iteration 2
Processing dependencies...
0 items added
Processing include files...
2 items added.
Processing exclude list...
0 items removed.
Processing installed directories...
548 items removed.
Generating MATLAB path...
Created 33 path items.
Depfun main loop converged in 2 iterations, total number of files = 8
Generating file "timings_mcc_component_data.c".
Executing command: mbuild -O -v -output 'timings' 'timings_main.c'
'timings mcc component data.c' -link exe
 This is mbuild Copyright 1984-2004 The MathWorks, Inc.
-> Default options filename found in C:\Documents and
Settings\jackchen\Application Data\MathWorks\MATLAB\R14
-----------------------------------------------------------------
-> Options file = C:\Documents and Settings\jackchen\Application Data\
MathWorks\MATLAB\R14\compopts.bat
->    COMPILER                = lcc
->    Compiler flags:
          COMPFLAGS               = -c -Zp8 -I"D:\SoftWare\MATLAB7.0
\sys\lcc\include" -noregistrylookup
         OPTIMFLAGS           = -DNDEBUG
         DEBUGFLAGS           = -g4
         arguments            =
         Name switch          = -Fo
->    Pre-linking commands    =
->    LINKER                  = lcclnk
->    Link directives:
         LINKFLAGS              = -tmpdir "." -L"D:\SoftWare\MATLAB7.0
\sys\lcc\lib" -libpath "D:\SoftWare\MATLAB7.0\extern\lib\win32\lcc"
         LINKFLAGSPOST        = mclmcrrt.lib
         Name directive       = -o "timings.exe"
         File link directive =
         Lib. link directive -
         Rsp file indicator   = @
->    Resource Compiler       =
->    Resource Linker         =
-----------------------------------------------------------------
--> "lcc  -c -Zp8 -I"D:\SoftWare\MATLAB7.0\sys\lcc\include" -
noregistrylookup -Fotimings_main.obj -ID:\SoftWare\MATLAB7.
0\extern\include -ID:\SoftWare\MATLAB7.0\simulink\include -DNDEBUG
timings_main.c"
--> "lcc  -c -Zp8 -I"D:\SoftWare\MATLAB7.0\sys\lcc\include" -
noregistrylookup -Fotimings_mcc_component_data.obj -ID:\SoftWare\MATLAB7.
0 \extern\include -ID:\SoftWare\MATLAB7.0\simulink\include -DNDEBUG
timings_mcc_component_data.c"
    Contents of 2541_tmp.rsp:
```

[illegible]

step 5 [illegible]

图 13.9 在 DOS 条件下执行可执行文件

13.5.2 编译 M 和 C 的混合文件

除了前面介绍的编译 M 文件外，MATLAB7.0 中提供的 Compiler 4.0 还可以编译 M 文件和 C 文件混合的文件。在本小节中，将以一个简单的例子来演示如何编译 M 和 C 的混合文件。

例 13.8 编译由 M 文件和 C 文件组成的混合文件。

step 1 简述文件组成部分。在本实例中，需要编译的文件如下：

- **mrank.m**：该 M 文件包含函数，该函数返回 1 到 n 维 magic 矩阵的秩。
- **mrankp.c**：在该 C 语言程序代码中，调用 mrank 文件中定义的函数，并返回该函数定义的输出参数数值。

step 2 查看 mrank.m 文件中的程序代码。该 M 文件包含的程序代码如下：

```
function r = mrank(n)
r = zeros(n,1);
for k = 1:n
    r(k) = rank(magic(k));
end
```

step 3 查看 mrankp.c 文件中的程序代码。该 C 文件中包含的程序代码如下：

```
#include <stdio.h>
#include <math.h>
#include "libPko.h"
main( int argc, char **argv )
{
    mxArray *N;          /* Matrix containing the */
    mxArray *R = NULL;   /* Result matrix */
    int     n;           /* Integer parameter from command line. */
```

```
/* 获取输入参数 */
if (argc >= 2) {
    n = atoi(argv[1]);
} else {
    n = 12;

mclInitializeApplication(NULL,0);
libPkgInitialize();/* Initialize the library of M-functions */
[illegible]
N = mxCreateScalarDouble(n);
/* 调用mlfMrank, mrank.m的编译版本 */
[illegible], N);
/* 显示结果 */
[illegible]);
/* 清空矩阵的数值 */
[illegible]);
mxDestroyArray(R);
libPkgTerminate(); /* Terminate the library of M-functions */
[illegible]Application();
```

Step 4 为M-[illegible]

- mxArray *N; … else n = 12 [illegible]要定义一个mlfMrank函数的输入参数。
- mclInitializeApplication(NULL,0) … libPkgInitialize() [illegible]M函数，并产生 libPkg 共享库。
- N = mxCreateScalarDouble(n); …mlfMrank(1, &R, N); [illegible]包含参数 n 的矩阵，然后调用编译版本的 mrank.m 函数。
- mlfPrintmatrix(R); … mclTerminateApplication(); [illegible]输出计算结果，[illegible]

[illegible]

Step 5 [illegible]

[illegible]

Step 6 运行可执行文件。在 DOS 环境中运行 mrank.exe 文件，如图 13.10 所示。

Step 7 [illegible]

```
>> mrank(12)
ans =

     7
     4
```

图 13.10 在DOS环境下运行文件

[illegible]

13.5.3 编译包含绘图命令的M文件

前面安装了MCR，因此MATLAB可以编译包含绘图命令的M文件。这是因为在安装MCR后，MATLAB会在编译过程中加入图形库。在本小节中，将以一个简单的实例来说明如何编译包含绘图命令的M文件。

例 13.9 编译Areoplot.m文件，该文件的功能主要是绘制在某表面体的图形非黏性、不旋转的不可以压缩的流体的流型线和压力系数图形。

Step 1 查看Areoplot.m文件的代码，其具体的程序代码如下

```
function Areoplot
%[illegible]系数数组
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]数
[illegible]
[illegible]:(b-c)/(n:b)]';
[illegible]
[illegible]
[illegible]
[illegible]
    [illegible]
        [illegible]
            [illegible]
            [illegible]
        [illegible]
```

```
    end
end
%定义极坐标的数值
rho=sqrt(x.^2+y.^2);
theta=atan2(y,x);
% 创建流线型的函数
z=V_i.*sin(theta).*rho.*(1-(a^2./(rho.^2)))-G*log(rho)/(2*pi);
% 创建图形
n=100;
r=ones(1,n+1)*a;
t=[ 0:2*pi/n:2*pi] ;
%流线型图形 contour(x,y,z,25)
hold on
polar(t,r,'-k')
axis square
title('Stream Lines')
grid off
figure(2)
contour(x,y,z,15)
%创建环绕在圆柱体的矢量场
x=[ -a*2:a/3:a*2] ;
[ x] =meshgrid(x);
y=x';
for i=1:length(x);
    for k=1:length(x);
        if sqrt(x(i,k).^2+y(i,k).^2)<a;
            x(i,k)=0;
            y(i,k)=0;
        end
    end
end
r=sqrt(x.^2+y.^2);
theta=atan2(y,x);
ur=V_i*cos(theta).*(1-a^2./(r.^2));
ut=-V_i*sin(theta).*(1+a^2./(r.^2))+G./(2*pi*r);
u=ur.*cos(theta)-ut.*sin(theta);
v=ur.*sin(theta)+ut.*cos(theta);
%创建填充后的图形
t_r  = 0:.1:2*pi;
xxx = a*cos(t_r);
yyy = a*sin(t_r);
%填充后的图形和矢量场图形
figure(2)
hold on
quiver(x,y,u,v)
fill(xxx,yyy,'y')
axis square
title('Speed Vectors')
grid off
warning on
t=0:.1:2*pi;
cp = 1 - 4*sin(t).^2 + 2* G / (pi*a*V_i) *sin(t) - (2* G/ (pi*a*V_i) )^2 ;
cp_sim = 1 - 4*sin(t).^2 ;
L = - 1.225*V_i*G;
L = num2str(L);
```

```
L = strcat('Kutta Joukowski Lift:  ',L,'  [N]');
figure1=
plot(t,cp,t,cp_sim,'--r')
axis([0 2*pi min(cp) max(cp_sim)])
title([L,'Pressure coefficient around the surface (standard air density)',
],
xlabel('Theta (angle with positive x-axis)')
ylabel('C_p')
legend('Lifting solution','Symmetrical solution')
grid on
```

[illegible]了，请读者自己思考

step 2 编辑Areoplot.m文件，在MATLAB的命令窗口中输入下面的编译命令

```
>>mcc -m Areoplot.m
```

step 3 查看所编译的结果，输入dir，按"Enter"键，得到结果如

```
Compiler version: 4.0 (R14)
Parsing file "d:\software\matlab7.0\work\areoplot.m"
    (Referenced from: "Compiler Command Line").
Parsing file "d:\software\matlab7.0\toolbox\compiler\deploy\matlabrc.m"
    (Referenced from: "Compiler Command Line").
Parsing file "d:\software\matlab7.0\toolbox\compiler\deploy\libmmfile.mlib"
    (Referenced from: "Compiler Command Line").
Parsing file "d:\software\matlab7.0\toolbox\matlab\graph2d\axis.m"
    (Referenced from: "d:\software\matlab7.0\work\areoplot.m").
Parsing file "d:\software\matlab7.0\toolbox\matlab\graph2d\title.m"
    (Referenced from: "d:\software\matlab7.0\work\areoplot.m").
Parsing file "d:\software\matlab7.0\toolbox\matlab\specgraph\private\xlabel.m"
    (Referenced from: "d:\software\matlab7.0\work\areoplot.m").

*****************************************************//省略了调用函数代码

Parsing file "d:\software\matlab7.0\toolbox\matlab\general\setenv.m"
    (Referenced from: "d:\software\matlab7.0\toolbox\compiler\
deploy\matlabrc.m").
Parsing file "d:\software\matlab7.0\toolbox\matlab\iofun\private\iterate.m"
    (Referenced from: "d:\software\matlab7.0\toolbox\compiler\deploy\iterate.m").

Generating file "areoplot_main.c".
Depfun main loop, iteration 1
Processing D:\SoftWare\MATLAB7.0\toolbox\matlab\mcc.enc
1 items added
Processing D:\SoftWare\MATLAB7.0\toolbox\database\mcc.enc
1 items added
Processing dependencies...
0 items added
Depfun main loop, iteration 2
Processing dependencies...
0 items added
Processing include files...
```

```
2 items added.
Processing exclude list...
0 items removed.
Processing installed directories...
1575 items removed.
Generating MATLAB path...
Created 34 path items.
Depfun main loop converged in 2 iterations, total number of files = 65
Generating file "areoplot_mcc_component_data.c".
Executing command: mbuild -O -v -output 'Areoplot' 'areoplot_main.c'
'areoplot_mcc component data.c' -link exe
 This is mbuild Copyright 1984-2004 The MathWorks, Inc.
-> Default options filename found in C:\Documents and
Settings\jackchen\Application Data\MathWorks\MATLAB\R14
-----------------------------------------------------------------
->      Options file                    = C:\Documents and
Settings\jackchen\Application Data\MathWorks\MATLAB\R14\compopts.bat
->    COMPILER                = lcc
->    Compiler flags:
        COMPFLAGS    = -c -Zp8 -I"D:\SoftWare\MATLAB7.0\sys\lcc\include"
-noregistrylookup
         OPTIMFLAGS           = -DNDEBUG
         DEBUGFLAGS           = -g4
         arguments            =
         Name switch          = -Fo
->    Pre-linking commands    =
->    LINKER                  = lcclnk
->    Link directives:
         LINKFLAGS             = -tmpdir "." -L"D:\SoftWare\MATLAB7.0
\sys\lcc\lib" -libpath "D:\SoftWare\MATLAB7.0\extern\lib\win32\lcc"
         LINKFLAGSPOST        = mclmcrrt.lib
         Name directive       = -o "Areoplot.exe"
         File link directive =
         Lib. link directive =
         Rsp file indicator   = @
->    Resource Compiler       =
->    Resource Linker         =
-----------------------------------------------------------------
--> "lcc  -c -Zp8 -I"D:\SoftWare\MATLAB7.0\sys\lcc\include" -
noregistrylookup -Foareoplot_main.obj -ID:\SoftWare\MATLAB7.0
\extern\include -ID:\SoftWare\MATLAB7.0\simulink\include -DNDEBUG
areoplot_main.c"
--> "lcc  -c -Zp8 -I"D:\SoftWare\MATLAB7.0\sys\lcc\include" -
noregistrylookup -Foareoplot_mcc_component_data.obj -ID:
\SoftWare\MATLAB7.0\extern\include -ID:\SoftWare\MATLAB7.0
\simulink\include -DNDEBUG areoplot_mcc_component_data.c"
    Contents of 5152_tmp.rsp:
  areoplot_main.obj  areoplot_mcc_component_data.obj
--> "lcclnk -o "Areoplot.exe" -tmpdir "." -L"D:\SoftWare\MATLAB7.0
\sys\lcc\lib"  libpath "D:\SoftWare\MATLAB7.0\extern\lib\win32\lcc"
@5152_tmp.rsp mclmcrrt.lib"
--> "if exist _lib5152.def del _lib5152.def"
--> "if exist %LIB_NAME_stub.obj del %LIB_NAME_stub.obj"
```

[illegible]
到文件和一般M文件有什么区别。

step 6 [illegible]

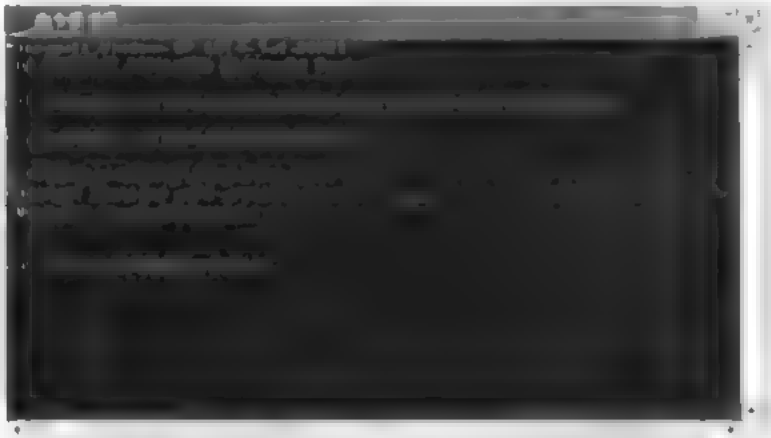


图 13.11 在 DOS 环境中运行可执行文件

[illegible]
户可以输入下一条命令。

[illegible]，MATLAB编译器[illegible]

图 13.12 程序绘制的直线型图形

[illegible]

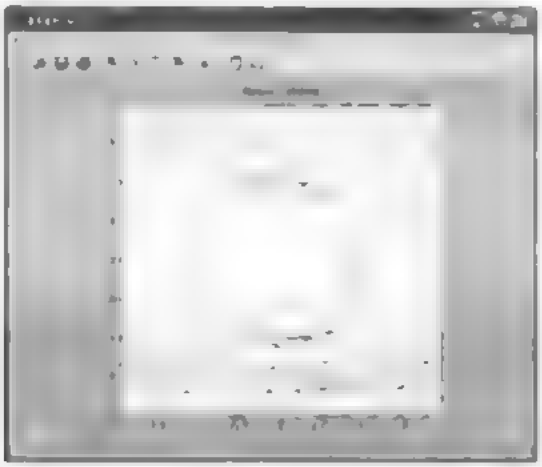


图 13.13　绘制的速度向量图形

[illegible]

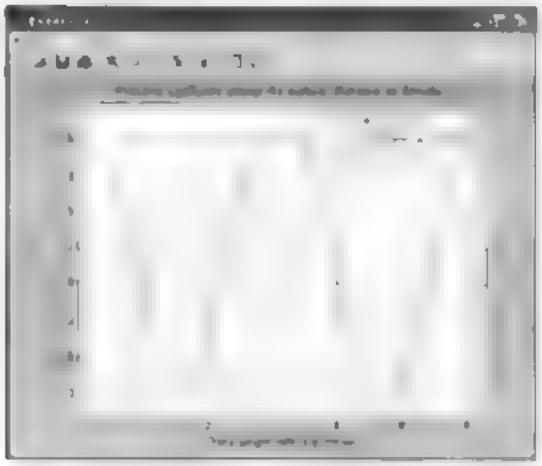


图 13.14　表面压力系数图形

step 5 对比图形界面结果。

[illegible]

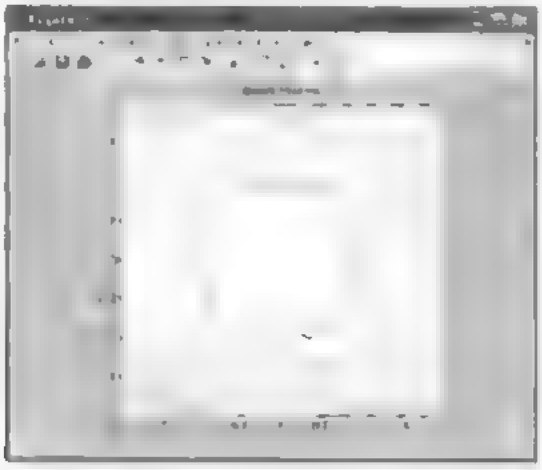


图 13.15　在 MATLAB 环境中得到的图形

小结

本章主要介绍了如何在MATLAB中安装和配置编译器，介绍了MATLAB中的编译命令和过程，最后还介绍了如何使用编译器来创建独立的应用程序，通过这些内容希望读者能够对MATLAB的编译器有大概的了解。下一章将介绍如何在MATLAB中编写应用程序接口。

第14章 应用程序接口

本章包括

- 使用C语言创建MEX文件
- 使用C语言创建MAT文件
- 使用FORTRAN语言创建MAT文件
- MATLAB的引擎技术
- Java接口

前面主要介绍了MATLAB自身的功能和使用方法，但是作为一个优秀工程软件，MATLAB除自身功能强大、环境友好之外，还有很好的开放性。这种开放性体现在MATLAB可以和外部应用程序实现“无缝”结合，提示了专业应用程序接口API。

在本章中主要从下面几个方面来介绍MATLAB应用程序接口。首先介绍MEX文件——外部程序调用接口，在MATLAB中使用C语言或者FORTRAN语言编写的程序代码，用来提供程序运行的效率；然后介绍MAT文件应用程序——数据输入输出接口，向MATLAB输入或者输出数据的程序代码。最后，将介绍MATLAB计算引擎函数库——在MATLAB和其他应用程序中建立客户机/服务器关系，将MATLAB作为计算引擎，在其他应用程序中调用，从而降低应用程序的计算量。在Windows操作系统中，MATLAB支持该系统提供的COM标准，同时支持Java语言，因此MATLAB几乎可以和任何一种软件进行交互。

C语言MEX文件

MEX文件是一种可以在MATLAB中调用的C语言或者FORTRAN语言衍生程序代码，而MEX本身就是MATLAB和Executable两个单词的缩写。通过C语言编写的MEX文件程序代码，经过适当的编译后，生成的目标文件能够被M语言解释器调用执行，在Windows操作系统下这些文件使用后缀dll(Dynamic link library)。MEX文件的使用极为方便，其调用方式与MATLAB的内建函数完全相同，只需要在命令窗口输入对应的文件名称即可。

MEX文件的数据

和其他语言程序代码的编写一样，在MATLAB中用户希望编写MEX程序代码，则首先有必要了解MEX文件中的数据类型，以及这些数据类型和MATLAB对应的数据之间的联系等。在本小节中，将简要介绍MEX中的主要数据类型。

由于在MATLAB中所有的数据都是以矩阵或者阵列(Array)存储的，因此如果使用C语言来编写MEX文件，就必须能够处理对应的数据类型。在C语言中，Array数据类型用mxArray来定义，这种结构体包含的信息有变量类型、维数和数据等。对于数值类型的变量，该结构体说明变量是实数还是复数；对于稀疏类型的变量，该结构体说明变量的下标和最大非零元素；对于构架类型的变量，该结构体说明变量的域名和对应的数值。

根据上面的介绍，该结构体的存储信息如下。

- **复数双精度数值矩阵：**该结构体将存储这些变量的双精度类属性、维数，双精度变量的实部变量和虚部变量，实部变量指针pr，虚部变量指针pi。

- **其他数值矩阵**：[illegible]方式和双精度矩阵相同。
- **字符串**：字符串类，每个字符串用16位ASCII Unicode码表示。
- **元胞数组变量**：每[illegible]包含了mxArray结构体的指针信息。
- **构架数据类型**：[illegible]体的mxArray结构体。

14.1.2 MEX文件的结构

[illegible]代码并不能直接编译成可以被MATLAB调用的MEX文件，必须经过专门的编译链接为MEX文件。在本小节中，将以一个简单的例子来说明MEX文件的结构。

例14.1 [illegible]

step 1 [illegible]代码

```
[illegible]

  y(0) = 2.0*x(0);
[illegible]
```

[illegible]

step 2 [illegible]应的代码如下

```
#include "mex.h"
void timestwo(double y[], double x[])
{
  y(0) = 2.0*x(0);

[illegible]
[illegible]
}
  double *x, *y;
  int mrows, ncols;

  /* 检测输入参数的个数 */
  if (nrhs != 1) {
    mexErrMsgTxt("One input required.");
  } else if (nlhs > 1) {
    mexErrMsgTxt("Too many output arguments");

  /* 确保输入参数是标量，并且是浮值*/
```

```
mrows = mxGetM(prhs[0]);
ncols = mxGetN(prhs[0]);
if (!mxIsDouble(prhs[0]) || mxIsComplex(prhs[0]) ||
    !(mrows == 1 && ncols == 1)) {
  mexErrMsgTxt("Input must be a noncomplex scalar double.");

/* 创建返回参数的属性 */
plhs[0] = mxCreateDoubleMatrix(mrows,ncols, mxREAL);
/* 为输入参数和输出参数分配指针 */
x = mxGetPr(prhs[0]);
y = mxGetPr(plhs[0]);
/* 调用子函数 */
```

step 3 [illegible]

```
>> mex timestwo.c
>> which timestwo.mexw64
```

[illegible]的文件信息如下

[illegible]

step 4 [illegible]

```
>> x = 2;
y = timestwo(x)
```

[illegible]

下面将以上最简单的例子来说明MEX的一般结构。

- #include "mex.h"…y[0] = 2.0*x[0]；[illegible]程序代码几乎没有差别。
- void mexFunction(int nlhs, mxArray *plhs[], int nrhs, const mxArray *prhs[])…mexErrMsgTxt("Too many output arguments")；[illegible]代码。然后检测程序代码输入参数的个数。
- mrows = mxGetM(prhs[0])…mexErrMsgTxt("Input must be a noncomplex scalar double.")；[illegible]并退出程序代码。

- plhs[0] = mxCreateDoubleMatrix(mrows,ncols, mxREAL)：[illegible]数创建存储的数组空间。
- x = mxGetPr(prhs[0]) …… y = mxGetPr(plhs[0])：[illegible]数分别指针变量。
- timestwo(y,x)：[illegible]

[illegible]

[illegible]

- [illegible]
- [illegible]
- [illegible]

[illegible]

在本小节的最后，要给读者一个程序代码上面的功能。该程序代码中使用一种特殊的API函数mxGetScalar，该函数将直接从数组中取值，而不需要使用指针变量。具体的程序代码如下

```
#include "mex.h"
void timestwo_alt(double *y, double x)
{
  *y = 2.0*x;

void mexFunction(int nlhs, mxArray *plhs[],
                 int nrhs, const mxArray *prhs[])
{
  double *y;
  double x;
  /* Create a 1-by-1 matrix for the return argument. */
  plhs[0] = mxCreateDoubleMatrix(1, 1, mxREAL);
  /* Get the scalar value of the input x. */
  /* Note: mxGetScalar returns a value, not a pointer. */
  x = mxGetScalar(prhs[0]);
  /* Assign a pointer to the output. */
  y = mxGetPr(plhs[0]);

  /* Call the timestwo_alt subroutine. */
  timestwo_alt(y,x);
```

[illegible]

14.1.3 MEX文件的实例

[illegible] MEX文件的理解。

例14.2 [illegible]

Step 1 [illegible]

```
 *
 * ------------------------------------------------------------------
 * phonebook.c
 * Takes a (MxN) structure matrix and returns a new structure
 * [illegible]
 * will be (MxN) cell array; and for numeric (noncomplex, scalar)
 * input, it will be (MxN) vector of numbers with the same
 * classID as input, such as int, double etc..
 *
 * ------------------------------------------------------------------
#include "mex.h"
#include "string.h"
#define MAXCHARS 80    /* max length of string contained in each
                          field */
/* The gateway 子程序 */
void mexFunction(int nlhs, mxArray *plhs[],
                 int nrhs, const mxArray *prhs[])
{
  const char **fnames;        /* 域名的指针变量 */
  const int  *dims;
  [illegible]
  char       *pdata;
  [illegible]
  [illegible]

  /* 确保正确的输入和输出变量*/
  if (nrhs != 1)
    mexErrMsgTxt("One input required.");
  else if (nlhs > 1)
    mexErrMsgTxt("Too many output arguments.");
  else if (!mxIsStruct(prhs[0]))
    mexErrMsgTxt("Input must be a structure.");
  /* 获取输入参数数值 */
  nfields = mxGetNumberOfFields(prhs[0]);
  NStructElems = mxGetNumberOfElements(prhs[0]);
  /* 为保存变量 classIDflags 分配内存 */
  classIDflags = mxCalloc(nfields, sizeof(int));
  [illegible]
  for (ifield = 0; ifield < nfields; ifield++) {
    for (jstruct = 0; jstruct < NStructElems; jstruct++) {
      tmp = mxGetFieldByNumber(prhs[0], jstruct, ifield);
      if (tmp == NULL) {
```

```
        mexPrintf("%s%d\t%s%d\n",
            "FIELD:", ifield+1, "STRUCT INDEX :", jstruct+1);
        mexErrMsgTxt("Above field is empty!");
      }
      if (jstruct == 0) {
        if ((!mxIsChar(tmp) && !mxIsNumeric(tmp)) ||
            mxIsSparse(tmp)) {
          mexPrintf("%s%d\t%s%d\n",
              "FIELD:", ifield+1, "STRUCT INDEX :", jstruct+1);
          mexErrMsgTxt("Above field must have either "
              "string or numeric non sparse data.");
        }
        classIDflags[ifield] = mxGetClassID(tmp);
      } else {
        if (mxGetClassID(tmp) != classIDflags[ifield]) {
          mexPrintf("%s%d\t%s%d\n",
              "FIELD:", ifield+1, "STRUCT INDEX :", jstruct+1);
          mexErrMsgTxt("Inconsistent data type in above field!");
        }
        else if (!mxIsChar(tmp) && ((mxIsComplex(tmp) ||
            mxGetNumberOfElements(tmp) != 1))) {
          mexPrintf("%s%d\t%s%d\n",
              "FIELD:", ifield+1, "STRUCT INDEX :", jstruct+1);
          mexErrMsgTxt("Numeric data in above field "
              "must be scalar and noncomplex!");
        }
      }
    }
  }
  /* 为保存指针变量分配内存空间 */
  fnames = mxCalloc(nfields, sizeof(*fnames));
  /* Get field name pointers */
  for (ifield = 0; ifield < nfields; ifield++) {
    fnames[ifield] = mxGetFieldNameByNumber(prhs[0],ifield);
  }
  /* 为输出变量创建结构体数组 */
  plhs[0] = mxCreateStructMatrix(1, 1, nfields, fnames);
  mxFree(fnames);
  ndim = mxGetNumberOfDimensions(prhs[0]);
  dims = mxGetDimensions(prhs[0]);
  for (ifield = 0; ifield < nfields; ifield++) {
    /* Create cell/numeric array */
    if (classIDflags[ifield] == mxCHAR_CLASS) {
      fout = mxCreateCellArray(ndim, dims);
    } else {
      fout = mxCreateNumericArray(ndim, dims,
          classIDflags[ifield], mxREAL);
      pdata = mxGetData(fout);
    }
    /* 从输入中复制数据 */
    for (jstruct = 0; jstruct < NStructElems; jstruct++) {
      tmp = mxGetFieldByNumber(prhs[0],jstruct,ifield);
      if (mxIsChar(tmp)) {
        mxSetCell(fout, jstruct, mxDuplicateArray(tmp));
      } else {
        size_t     sizebuf;
```

```
        sizebuf = mxGetElementSize(tmp);
        memcpy(pdata, mxGetData(tmp), sizebuf);
        pdata += sizebuf;


    /* 设置输出结构的名称 */
    [illegible]


  [illegible]
  [illegible]
}
```

[illegible]和到 MATLAB 的用户工作路径中。

Step 2 [illegible]

```
>>mex phonebook.c
>> dir phonebook.*
```

Step 3 [illegible]

```
phonebook.c     phonebook.dll
```

[illegible]

Step 4 运行程序代码。在 MATLAB 的命令窗口中输入下面的程序代码

```
friends(1).name = 'Jordan Robert';
friends(1).phone = 3386;
friends(2).name = 'Mary Smith';
friends(2).phone = 3912;
friends(3).name = 'Stacy Flora';
friends(3).phone = 3238;
friends(4).name = 'Harry Alpert';
friends(4).phone = 3077;
phonebook(friends)
```

Step 5 [illegible]

```
ans =

    name: {'Jordan Robert'  'Mary Smith'  'Stacy Flora'  'Harry Alpert'}
   phone: [3386  3912  3238  3077]
```

[illegible]

例 14.3 [illegible]

Step 1 [illegible]

```
/*
 * ==========================================================
 * sincall.c
 *
 *
 * ==========================================================
#include "mex.h"
#define MAX 1000
/* Subroutine for filling up data */
void fill(double *pr, int *pm, int *pn, int max)
{
  int i;
  /* You can fill up to max elements, so (*pr) <- max. */
  *pm = max/2;
  *pn = 1;
  for (i = 0; i < (*pm); i++)
    pr[i] = i * (4*3.14159/max);
}
/* gateway子程序 */
void mexFunction(int nlhs, mxArray *plhs[],
                 int nrhs, const mxArray *prhs[])
{
  int     m, n, max = MAX;
  mxArray *rhs[1], *lhs[1];
  rhs[0] = mxCreateDoubleMatrix(max, 1, mxREAL);
  /* 传递指针变量数组，并填充数组数值 */
  fill(mxGetPr(rhs[0]), &m, &n, MAX);
  mxSetM(rhs[0], m);
  mxSetN(rhs[0], n);
  /* 获取sine曲线数据，并绘制该曲线 */
  mexCallMATLAB(1, lhs, 1, rhs, "sin");
  mexCallMATLAB(0, NULL, 1, lhs, "plot");
  /* 清除分配的内存空间 */
  mxDestroyArray(rhs[0]);
  mxDestroyArray(lhs[0]);
return;
}
```

输入上面的程序代码后，将上面的程序代码保存为 sincall.c，然后将该程序代码文件保存到 MATLAB 的目录路径中。

step 2 编译并运行程序代码。在 MATLAB 的命令窗口中输入下面的程序代码：

```
>> mex sincall.c
>> sincall
```

step 3 查看图形结果。输入代码后，按 "Enter" 键，得到的图形如图 14.1 所示。

在 C 语言编写的 MEX 程序代码中，用户可以使用 API 函数 mexCallMATLAB 来调用 MATLAB 中的函数、运算符、M 文件或者其他 MEX 文件。在上面的实例中，首先定义了变量 mxArray，同时通过不同的指针变量来传递数据，最后使用 mexCallMATLAB 来调用 sine 和 plot 函数来计算和绘制图形结果。

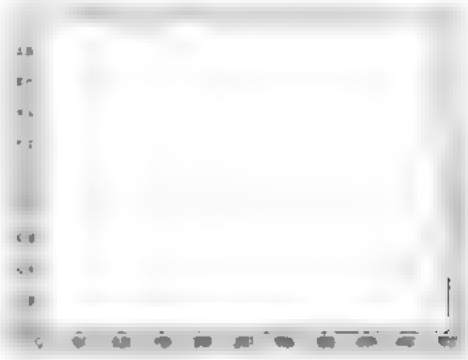


图 14.1 调用程序得到的结果

说明 在MATLAB中，会要求用户安装了FORTRAN开发环境，例如Compaq Visual Fortran等，[illegible] FORTRAN语言编写的MEX文件 [illegible] [illegible]。[illegible]

14.2 MAT 文件

MAT文件是用于在MATLAB和其他应用程序之间进行数据交换的重要方式和手段。在MATLAB中，MAT文件是使用C语言或者FORTRAN语言编写的一种文件，它可以是MEX文件，也可以是独立的[illegible]程序。为了便于操作MAT文件，MATLAB提供了相应的接口函数MAT函数。MAT文件应用程序就是利用这些MAT函数来完成MAT数据文件操作的文件。本节将介绍如何使用C语言和FORTRAN语言编写MEX文件。

14.2.1 使用C语言创建MAT文件

在本小节中，将使用简单的示例来介绍如何使用C语言来创建MAT文件，希望读者从中能够了解MAT应用程序的基本结构和应用的过程。

例 14.4 使用C语言编写创建MAT文件的程序代码。

step 1 打开用户在系统中安装的开发环境，创建C语言开发项目，然后在该开发项目中输入下面的程序代码

```
/*
 * MAT-file creation program
 *
 *
 * This program demonstrates the use of the following functions:
 *
 *  matClose
 *  matGetVariable
 *  matOpen
 *  matPutVariable
 *  matPutVariableAsGlobal
 *
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
```

```
#include "mat.h"
#define BUFSIZE 256
int main() {
%为MAT文件定义指针
  MATFile *pmat;
%定义结构体的指针
  mxArray *pa1, *pa2, *pa3;
%定义双精度变量
  double data[9] = { 1.0, 4.0, 7.0, 2.0, 5.0, 8.0, 3.0, 6.0, 9.0 };
  const char *file = "mattest.mat";
%定义字符串变量
  char str[BUFSIZE];
  int status;
%以标准的C格式输出MAT文件名称
  printf("Creating file %s...\n\n", file);
%以"写"模式打开名称为file的MAT文件
  pmat = matOpen(file, "w");
  if (pmat == NULL) {
    printf("Error creating file %s\n", file);
    printf("(Do you have write permission in this directory?)\n");
    return(EXIT_FAILURE);
  }
  pa1 = mxCreateDoubleMatrix(3,3,mxREAL);
  if (pa1 == NULL) {
      printf("%s : Out of memory on line %d\n", __FILE__, __LINE__);
      printf("Unable to create mxArray.\n");
      return(EXIT_FAILURE);
  }
  pa2 = mxCreateDoubleMatrix(3,3,mxREAL);
  if (pa2 == NULL) {
      printf("%s : Out of memory on line %d\n", __FILE__, __LINE__);
      printf("Unable to create mxArray.\n");
      return(EXIT_FAILURE);
  }
%将data缓冲区中的内容复制到pa2所指实部的目标缓冲区中
  memcpy((void *)(mxGetPr(pa2)), (void *)data, sizeof(data));
%为pa3所创建字符串的指针
  pa3 = mxCreateString("MATLAB: the language of technical computing");
  if (pa3 == NULL) {
      printf("%s :  Out of memory on line %d\n", __FILE__, __LINE__);
      printf("Unable to create string mxArray.\n");
      return(EXIT_FAILURE);
  }
  status = matPutVariable(pmat, "LocalDouble", pa1);
  if (status != 0) {
      printf("%s :  Error using matPutVariable on line %d\n", __FILE__,
__LINE__);
      return(EXIT_FAILURE);
  }
  status = matPutVariableAsGlobal(pmat, "GlobalDouble", pa2);
  if (status != 0) {
      printf("Error using matPutVariableAsGlobal\n");
      return(EXIT_FAILURE);
  }
  status = matPutVariable(pmat, "LocalString", pa3);
  if (status != 0) {
```

```
        printf("%s :  Error using matPutVariable on line %d\n", __FILE__,
__LINE__);
        return(EXIT_FAILURE);
    }
    /*

    memcpy((void *)(mxGetPr(pa1)), (void *)data, sizeof(data));
    status = matPutVariable(pmat, "LocalDouble", pa1);
    if (status != 0) {
        printf("%s :  Error using matPutVariable on line %d\n", __FILE__,
__LINE__);
        return(EXIT_FAILURE);
    }
    /* 释放所有的内存空间*/
    mxDestroyArray(pa1);
    mxDestroyArray(pa2);
    mxDestroyArray(pa3);
    if (matClose(pmat) != 0) {
      printf("Error closing file %s\n",file);
      return(EXIT_FAILURE);
    }
    /*
     * 再次打开MAT文件，对写入的内容进行验证
     */
    pmat = matOpen(file, "r");
    if (pmat == NULL) {
      printf("Error reopening file %s\n", file);
      return(EXIT_FAILURE);
    }
    /*
     * 读入之前定义的所有数据行
     */
    pa1 = matGetVariable(pmat, "LocalDouble");
    if (pa1 == NULL) {
      printf("Error reading existing matrix LocalDouble\n");
      return(EXIT_FAILURE);
    }
    if (mxGetNumberOfDimensions(pa1) != 2) {
     printf("Error saving matrix: result does not have two dimensions\n");
      return(EXIT_FAILURE);
    }
    pa2 = matGetVariable(pmat, "GlobalDouble");
    if (pa2 == NULL) {
      printf("Error reading existing matrix GlobalDouble\n");
      return(EXIT_FAILURE);
    }
    if (!(mxIsFromGlobalWS(pa2))) {
      printf("Error saving global matrix: result is not global\n");
      return(EXIT_FAILURE);
    }
    pa3 = matGetVariable(pmat, "LocalString");
    if (pa3 == NULL) {
      printf("Error reading existing matrix LocalString\n");
      return(EXIT_FAILURE);
    }
    status = mxGetString(pa3, str, sizeof(str));
```

```
  if(status != 0) {
      printf("Not enough space. String is truncated.");
      [illegible]
  }
  if (strcmp([illegible], "MATLAB: the language of technical computing")) {
    printf("Error saving string: result has [illegible] conversion");
    return(EXIT_FAILURE);

  /* 释放所有的内存空间 */
  [illegible]
  mxDestroyArray(pa2);
  [illegible]
  if (matClose(pmat) != 0) {
    [illegible]
    [illegible]
  }
  printf("Done\n");
  return(EXIT_SUCCESS);
}
```

在文件菜单中选择保存，将上面程序代码保存为 matlab.exe 文件。

step 2 编译上面程序代码，在 MATLAB 命令窗口中输入上面程序代码

```
[illegible] -f [illegible]
```

上面的编译代码中，mexoptdemo.bat 是默认的编译器的选项，在本章的开始时已经安装了 Microsoft Visual C++ 6.0 后，需要修改 mexopts.bat 文件，可以根据自己的编译器情况来选择对应的编译器选项。

step 3 运行上面程序代码，上面程序代码将在当前工作目录路径下创建 mattest.mat 文件，为了方便查看当前路径下的文件，在 MATLAB 的命令窗口中输入下面程序代码

```
whos -file mattest.mat
  Name              Size          Bytes  Class
  GlobalDouble      3x3              72  double array (global)
  LocalDouble       3x3              72  double array
  LocalString       1x43             86  char array

Grand total is 61 elements using 230 bytes
```

step 4 查看变量的内容，在 MATLAB 的命令窗口中输入变量名称，查看其内容，内容如下

```
GlobalDouble =
     1     2     3
     4     5     6
     7     8     9
LocalDouble =
     1     2     3
     4     5     6
     7     8     9
```

```
LocalString =
MATLAB: the language of technical computing
```

[illegible] MAT 文件 [illegible] MAT 文件 [illegible]

14.2.2 使用 FORTRAN 语言创建 MAT 文件

[illegible] 语言来创建 MAT 文件。

例 14.5 使用 C 语言编写创建 MAT 文件的程序代码。

step 1 [illegible]

然后在开发工具中输入下面的程序代码

```
C       matdemo1.f
        This is a simple program that illustrates how to call the
C       MATLAB MAT-file functions from a Fortran program.  This
C       demonstration focuses on writing MAT-files.

C       matdemo1 - Create a new MAT-file from scratch.
        [illegible]
        integer matOpen, matClose
        integer matGetVariable, matPutVariable
        [illegible]
        integer mxCreateDoubleMatrix, mxCreateString
        integer mxIsFromGlobalWS, mxGetPr
        integer mp, pa1, pa2, pa3, pa0, status
        double precision dat(9)
        data dat / 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0 /
C       Open MAT-file for writing.
        write(6,*) 'Creating MAT-file matdemo.mat ...'
        mp = matOpen('matdemo.mat', 'w')
        if (mp .eq. 0) then
           [illegible]
           write(6,*) '(Do you have write permission in this
                       directory?)'
           stop
        [illegible]
        创建变量
        pa0 = mxCreateDoubleMatrix(3,3,0)
        call mxCopyReal8ToPtr(dat, mxGetPr(pa0), 9)
        pa1 = mxCreateDoubleMatrix(3,3,0)
        pa2 = mxCreateString('MATLAB: The language of computing')
        pa3 = mxCreateString('MATLAB: The language of computing')
        status = matPutVariableAsGlobal(mp, 'NumericGlobal', pa1)
        if (status .ne. 0) then
           write(6,*) 'matPutVariableAsGlobal ''Numeric Global''
                      failed'
           stop
        end if
        status = matPutVariable(mp, 'Numeric', pa1)
```

```
      if (status .ne. 0) then
         write(6,*) 'matPutVariable ''Numeric'' failed'
         stop
      end if
      status = matPutVariable(mp, 'String', pa2)
      if (status .ne. 0) then
         write(6,*) 'matPutVariable ''String'' failed'
         stop
      end if
      status = matPutVariable(mp, 'String2', pa3)
      if (status .ne. 0) then
         write(6,*) 'matPutVariable ''String2'' failed'
         stop
      end if
C
      call mxCopyReal8ToPtr(dat, mxGetPr(pa1), 9)
      status = matPutVariable(mp, 'Numeric', pa1)
      if (status .ne. 0) then
         write(6,*) 'matPutVariable ''Numeric'' failed 2nd time'
         stop
      end if
C     从MAT文件中删除String2 变量
      status = matDeleteVariable(mp, 'String2')
      if (status .ne. 0) then
         write(6,*) 'matDeleteVariable ''String2'' failed'
         stop
      end if
C     重新阅读MAT文件
      status = matClose(mp)
      if (status .ne. 0) then
         write(6,*) 'Error closing MAT-file'
         stop
      end if
      mp = matOpen('matdemo.mat', 'r')
      if (mp .eq. 0) then
         write(6,*) 'Can''t open ''matdemo.mat'' for reading.'
         stop
      end if
      pa0 = matGetVariable(mp, 'NumericGlobal')
      if (mxIsFromGlobalWS(pa0) .eq. 0) then
         write(6,*) 'Invalid non-global matrix written to MAT-file'
         stop
      end if
      pa1 = matGetVariable(mp, 'Numeric')
      if (mxIsNumeric(pa1) .eq. 0) then
         write(6,*) 'Invalid non-numeric matrix written to
                    MAT-file'
         stop
      end if
      pa2 = matGetVariable(mp, 'String')
      if (mxIsString(pa2) .eq. 0) then
         write(6,*) 'Invalid non-string matrix written to MAT-file'
         stop
      end if
      pa3 = matGetVariable(mp, 'String2')
      if (pa3 .ne. 0) then
```

```
            write(6,*) 'String2 not deleted from MAT-file'
            stop
         end if
         [illegible]
         call mxDestroyArray(pa0)
         call mxDestroyArray(pa1)
         call mxDestroyArray(pa2)
         call mxDestroyArray(pa3)
         status = matClose(mp)
         if (status .ne. 0) then
            [illegible] MAT-file'
            stop
         end if
         [illegible] MAT-file'
         stop
         [illegible]
```

不输入；在文件名后[illegible]，将生成的目标文件matdemo.dll，然后将该文件保存在用户使用MATLAB的目标路径中。

Step 2 编译[illegible]，在MATLAB主窗口中输入[illegible]，且[illegible]。

```
>>mex matdemo1.f
```

[illegible] MAT文件matdemo.mat，[illegible]MATLAB工作数据输入[illegible]MAT文件[illegible]，变量名称为matdemo.dll，保存[illegible]。

```
Creating MAT-file matdemo.mat ...
Done creating MAT-file
```

Step 3 命令行[illegible]，在MATLAB主窗口中输入[illegible]，则[illegible]，内容[illegible]如下

```
Name          Size          Bytes  Class
 Numeric       3x3             72  double array
 [illegible]

Grand total is 42 elements using 138 bytes
```

Step 4 查看变量[illegible]，在主窗口中输入变量名称，则[illegible]

```
Numeric =
     1     2     3
     4     5     6
     7     8     9
String =
MATLAB: The language of computing
```

除了可以编译创建MEX文件[illegible]，在MATLAB中可以[illegible]FORTRAN语言来编写[illegible]取MAT文件中的MEX文件或者[illegible]程序，[illegible]，[illegible]时

MATLAB 引擎技术

前面花费了一定的篇幅来介绍MEX文件，本节将会介绍另外一种和该文件思想完全相反的内容——MATLAB引擎技术，也就是在其他应用程序中调用MATLAB的程序，例如调用MATLAB的Math库，进行数值计算等。

引擎技术概念

MATLAB中拥有一个引擎库。在该引擎库中汇集了多种函数，用户可以在自行编写的程序代码中引用这些函数，实现对MATLAB的调用。也就是说，用户可以自行编写界面运行在前台，而MATLAB作为计算引擎后台。引擎函数本身是使用C语言或者FORTRAN编写的，在Windows平台中，它和MATLAB之间的通信是通过ActiveX实现的。MATLAB引擎可以运用在下面的场合中：

- MATLAB在由其他语言编写的应用程序中被当作数学库程序调用，这样就可以在其他应用程序中利用MATLAB命令简单、计算可靠的优点；
- MATLAB在专门系统中当作计算引擎使用时，前台是其他应用程序语言所编写的GUI图形接口，后台由MATLAB来进行计算，这样就可以节省用户的开发时间。

引擎技术应用

在本小节中将利用一个简单的实例来介绍如何使用C语言编写程序代码，在该程序代码中调用MATLAB计算引擎。

例 14.6 使用C语言编写引擎应用的实例。

step 1 打开用户系统中安装的C语言开发工具，然后在开发工具中输入下面的程序代码：

```
/*
 *  engwindemo.c
#include <windows.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "engine.h"
#define BUFSIZE 256
static double Areal[6] = { 1, 2, 3, 4, 5, 6 };
int PASCAL WinMain (HINSTANCE hInstance,
                    HINSTANCE hPrevInstance,
                    LPSTR     lpszCmdLine,
                    int       nCmdShow)
{
%定义ep为MATLAB引擎的指针
    Engine *ep;
%定义三个空的结构体
    mxArray *T = NULL, *a = NULL, *d = NULL;
%定义容量为257的缓冲区
    char buffer[BUFSIZE+1];
%定义双精度变量的指针
    double *Dreal, *Dimag;
%定义双精度变量
    double time[10] = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };
```

```
    /*
     * 启动MATLAB引擎，如果出现错误则退出程序
     */
    if (!(ep = engOpen(NULL))) {
          MessageBox ((HWND)NULL, (LPSTR)"Can't start MATLAB engine",
                (LPSTR) "Engwindemo.c", MB_OK);
          exit( 1);
    }
    /*
     * 创建变量
    T = mxCreateDoubleMatrix(1, 10, mxREAL);
    memcpy((char *) mxGetPr(T), (char *) time, 10*sizeof(double));
    /*
     * 将变量T传递到MATLAB的工作空间中
    engPutVariable(ep, "T", T);
    /*
     * 根据公式 distance = (1/2)g.*t.^2计算变量数值
          engEvalString(ep, "D = .5.*(-9.8).*T.^2;");
    / * 绘制计算结果
    engEvalString(ep, "plot(T,D);");
    engEvalString(ep, "title('Position vs. Time for a falling object');");
    engEvalString(ep, "xlabel('Time (seconds)');");
    engEvalString(ep, "ylabel('Position (meters)');");
     /* 计算特征值数值
           engEvalString(ep, "d = eig(A*A')");
     /*
     * 获取MATLAB的输出变量
     buffer[BUFSIZE] = '\0';
     engOutputBuffer(ep, buffer, BUFSIZE);
     /*
     * 返回计算数值到缓冲区中
     engEvalString(ep, "whos");
     MessageBox ((HWND)NULL, (LPSTR)buffer, (LPSTR) "MATLAB - whos",
MB_OK);
     /*
     * 计算特征值数据矩阵
     d = engGetVariable(ep, "d");
%关闭ep所指向的引擎
     engClose(ep);
     if (d == NULL) {
                MessageBox ((HWND)NULL, (LPSTR)"Get Array Failed", (LPSTR)
"Engwindemo.c", MB_OK);
          }
    else {
          Dreal = mxGetPr(d);
          Dimag = mxGetPi(d);
          if (Dimag)
                sprintf(buffer,"Eigenval 2: %g+%gi",Dreal[1],Dimag[1]);
          else
                sprintf(buffer,"Eigenval 2: %g",Dreal[1]);
          MessageBox ((HWND)NULL, (LPSTR)buffer, (LPSTR)"Engwindemo.c",
MB_OK);
        mxDestroyArray(d);
    }
    /* 释放所有的内存空间
    mxDestroyArray(T);
```

```
    mxDestroyArray(A);
    return(0);
}
```

[illegible]，将程序保存为"engwindemo.c"文件，并将该文件保存在用户使用MATLAB的目录路径中。

step 2 编译：在命令窗口，在MATLAB主命令窗口中输入下面的命令行

```
>> mex -f [illegible] -output engwindemo [illegible] engwindemo.c
```

step 3 运行编译文件：输入命令后，按"Enter"键，在当前的目录路径中会生成[illegible]文件，[illegible]在MATLAB的命令窗口中输入下面的命令行

```
>> !engwindemo6
```

step 4 查看程序的运行结果：输入命令后，按"Enter"键，[illegible]

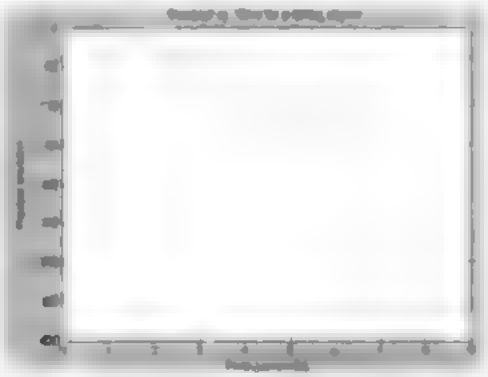


图 14.2 计算结果图形

在此，计算结果是开发平台中，MATLAB还会产生一个提示，这提示信息会在MATLAB的命令窗口，如图 14.3所示。

图 14.3 在运行程序时显示的对话框

step 5 查看程序变量的结果。在命令窗口中查看程序运行后的变量结果

```
>> whos
  Name      Size                    Bytes  Class
  A         3x2                        48  double array
  [illegible]  1x10                       80  double array
  T         1x10                       80  double array
  d         3x1                        24  double array

Grand total is 29 elements using 232 bytes
>>
```

```
A

        4

        0
        1
        2
        3
        4
        5
        6
        7
        8
        9

   -0.0000    0.5973   90.4027
```

最后，根据不同的需求，选择查看全部变量信息的对话框，如图 14.4 所示。

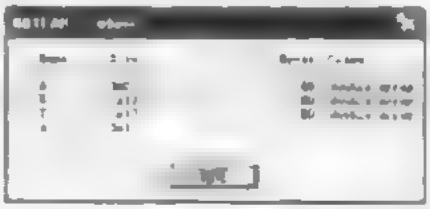


图 14.4　程序变量信息对话框

在使用 MATLAB 编程，管理内存，是的确不容忽视的问题。编写程序时，要注意以下几点：

- 首先，在开发过程中的初期，在代码的最开始部分加上一段清除所有文件、内存和图形句柄的语句，名称为 clear all，这样，在文件中调用了有 clear 函数的函数时，
- 在程序结束时，一般要释放掉所有的指针，或将计算结果写入文件以便于后续处理，
- 在程序结束时，一般要关闭计算引擎，这可以通过调用函数 close 来实现，通过这条命令可以关闭指针来释放内存。
- MATLAB 的计算引擎是基于客户机/服务器结构的计算引擎，设置数据、执行 MATLAB 的命令、获取计算结果，最后关闭计算引擎。

[illegible]

14.4 Java 接口

[illegible]

[illegible]

14.4.1 Java 接口语言基础

在 MATLAB 中使用 Java 语言之前，首先有必要了解当前 MATLAB 所使用的 Java 语言的版本，用户可以使用 "version -java" 命令得到版本信息，如下：

```
>> version -java
ans =
Java 1.4.2 with Sun Microsystems Inc. Java HotSpot(TM) Client VM
    (mixed mode)
```

根据上述的版本信息，用户可以[illegible]

Java 是一种面向对象的高级编程语言，[illegible]

在 MATLAB 中，[illegible]

```
>> [illegible]
        STATIC JAVA PATH

D:\SoftWare\MATLAB7.0\java\jar\[illegible]
D:\SoftWare\MATLAB7.0\java\jar\[illegible]
D:\SoftWare\MATLAB7.0\java\jar\[illegible]
D:\SoftWare\MATLAB7.0\java\jar\beans.jar
D:\SoftWare\MATLAB7.0\java\jar\[illegible]
D:\SoftWare\MATLAB7.0\java\jar\[illegible]
D:\SoftWare\MATLAB7.0\java\jar\ide.jar
D:\SoftWare\MATLAB7.0\java\jar\jmi.jar
D:\SoftWare\MATLAB7.0\java\jar\[illegible]
D:\SoftWare\MATLAB7.0\java\jar\[illegible]
D:\SoftWare\MATLAB7.0\java\jar\miwidgets.jar
D:\SoftWare\MATLAB7.0\java\jar\[illegible]
[illegible]
D:\SoftWare\MATLAB7.0\java\[illegible]
```

```
D:\SoftWare\MATLAB7.0\java\jarext\jaccess-1_4.jar
D:\SoftWare\MATLAB7.0\java\jarext\junit.jar
D:\SoftWare\MATLAB7.0\java\jarext\mwucarunits.jar
D:\SoftWare\MATLAB7.0\java\jarext\saxon.jar
D:\SoftWare\MATLAB7.0\java\jarext\vb20.jar
D:\SoftWare\MATLAB7.0\java\jarext\wsdl4j.jar
D:\SoftWare\MATLAB7.0\java\jarext\xalan.jar
D:\SoftWare\MATLAB7.0\java\jarext\xercesImpl.jar
D:\SoftWare\MATLAB7.0\java\jarext\xml-apis.jar

      DYNAMIC JAVA PATH
<empty>
```

从上面的结果中可以看出，在默认情况下将分为静态和动态Java路径，其中静态路径主要用来保存稳定、静态的Java类，而对于需要编辑的Java类，则建议保存在动态路径中。在默认情况下，classpath.txt文件会保存在toolbox\local路径下，具体信息如下：

```
>> which classpath.txt
D:\SoftWare\MATLAB7.0\toolbox\local\classpath.txt
```

在默认情况下，MATLAB本身会自动加载Java的内置函数类。为了查看当前使用的MATLAB中加载的所有函数类名，可以使用inmem命令查看所有的类名：

```
>> [m,x,j] = inmem
m =
    'matlabrc'
    'pathdef'
    'userpath'
    'ispc'
    'filesep'
    'pwd'
    'usejava'
    'hgrc'
    'opaque.char'
    'colordef'
    'whitebg'
    'jet'
    'initprefs'
    'findallwinclasses'
    'initdesktoputils'
    'path'
    'mdbstatus'
    'workspacefunc'
    'num2str'
    'mat2str'
    'int2str'
    'strvcat'
    'javaclasspath'
    'pathsep'
    'iscellstr'
x =
    'cellfun'
j =
    'java.util.Locale'
    'GObject'
```

```
'schema.class'
''[illegible]'
'schema.method'
'java.lang.String'
'[illegible]'
'[illegible]'
```

在上述的例子中，由于[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]。

在 MATLAB 中，可以直接使用 java 类定义和 javaObject 函数来创建 Java 对象，这两种使用方法都比较简单，下面使用实例来简要说明。

例 14.7 在 MATLAB 中创建 Java 对象。

step 1 创建 Java 对象。在 MATLAB 命令窗口中输入下列语句和命令

```
>> url = java.net.URL(...
    '[illegible]');
frame = java.awt.Frame('Frame Java B');
class = 'java.lang.String';
text = 'hello';
strObj = javaObject(class, text);
origFrame = java.awt.Frame;
[illegible]
newFrameRef = origFrame;
[illegible]
getSize(origFrame);
```

step 2 查看[illegible]。在命令窗口中输入 'whos'，[illegible]，[illegible]。

```
>> whos
  Name          Size          Bytes  Class
  ans           1x1                  java.awt.Dimension
  class         1x16            32   char array
  frame         1x1                  java.awt.Frame
  newFrameRef   1x1                  java.awt.Frame
  [illegible]   [illegible]          [illegible]
  strObj        1x1                  java.lang.String
  text          1x5             10   char array
  url           1x1                  java.net.URL

Grand total is 27 elements using 42 bytes
```

step 3 查看 Java 对象。在 MATLAB 命令窗口中，[illegible]。

```
[illegible]
url =
[illegible]
>> frame
frame =
[illegible]
BorderLaycut,title=Frame Java B,resizable,normal]
```

```
>> strObj
 strObj =
hello
[illegible]
newFrameRef =
[illegible]
>> getSize(origFrame)
ans =
java.awt.Dimension[width=1000,height=800]
```

[illegible]个，[illegible]Java内置函数和Java的对象，[illegible]在例子中了解在MATLAB中构建Java对象的方法。

例 14.8 在 MATLAB 中对 Java 对象进行操作。

step 1 [illegible]

```
>> point1 = java.awt.Point(24,127);
point2 = java.awt.Point(114,29);
>> point=cat(1, point1, point2);
>> byte = java.lang.Byte(127);
integer = java.lang.Integer(52);
double = java.lang.Double(7.8);
>> number=[byte; integer; double];
```

step 2 [illegible]

```
>> point
point =
java.awt.Point[]:
    [ java.awt.Point]
    [ java.awt.Point]
>> number
number =
 java.lang.Number[]:
    [     127]
    [      52]
    [ 7.8000]
```

在上面的例子中，[illegible]Java定义了Java对象，[illegible]了对象中不同的变量进行选择。

例 14.9 在 MATLAB 中改变 Java 对象的属性。

step 1 [illegible]

```
>> frame=java.awt.Frame('A')
frame =
[illegible]
BorderLayout,title=A,resizable,normal]
>> setTitle(frame, 'Sample Frame')
>> frame
```

```
frame =
[illegible]
>> title = getTitle(frame)
title =
Sample Frame
```

在上面的程序中，使用[illegible]命令设置frame对象的属性，然后使用getTitle命令来获取该对象的属性。

step 2 使用invokeMethod()函数设置java对象的属性。在命令窗口中输入下面的程序代码。

```
>> gAddress = java.lang.String('Four score and seven years ago');
str = java.lang.String('Four score');
[illegible]
ans =
[illegible]
```

在上面的程序中，使用的java类型是String，创建gAddress字符串对象和名为"str"字符串对象的语句不同。执行命令后，会显示结果正确。这种情况下，gAddress字符串对象的确以"Four"字符串对象结尾。

例 14.10 在 MATLAB 中创建 Java 类型的数组。

step 1 创建Java数据类型的数组。在MATLAB命令窗口中输入下面的程序代码。

```
>> origArray = javaArray('java.lang.Double', 3, 4);
for m = 1:3
   for n = 1:4
      origArray(m,n) = java.lang.Double((m * 10) + n);
   end
end
>> origArray
```

step 2 查看程序的结果。输入代码后，按"Enter"键，显示如下的结果。

```
origArray =
[illegible]
    [11]    [12]    [13]    [14]
    [21]    [22]    [23]    [24]
    [31]    [32]    [33]    [34]
```

在上面的程序中，首先使用[illegible]命令创建了一个[illegible]的java数值数组，接下来该数组中的数值分别为java.lang.Double，然后使用循环语句来添加数组的数值，最终显示出结果。

例 14.11 在 MATLAB 中，将 Java 类型的数据转换为 MATLAB 中的一维数组。

step 1 在MATLAB的命令窗口中输入下面的程序代码。

```
>> import java.lang.* java.awt.*;
% Create a Java array of double
```

```
[illegible]
[illegible]
    dblArray(1, m) = Double(m * 7);
end
% Create a Java array of points
ptArray = javaArray('java.awt.Point', 3);
ptArray(1) = Point(7.1, 22);
ptArray(2) = Point(5.2, 35);
ptArray(3) = Point(3.1, 49);
% Create a Java array of strings
[illegible]
strArray(1,1) = String('one');    strArray(1,2) = String('two');
[illegible]
% Convert each to cell arrays
cellArray = {cell(dblArray), cell(ptArray), cell(strArray)}
```

step 2 [illegible]

```
cellArray =
    {1x10 cell}    {3x1 cell}    {2x2 cell}
>> cellArray(1,1)
ans =
    [ 7]    [ 14]    [ 21]    [ 28]    [ 35]    [ 42]    [ 49]    [ 56]    [ 63]
[ 70]
>> cellArray(1,2)
ans =
    [illegible]
    {1x1 java.awt.Point}
    {1x1 java.awt.Point}
>> cellArray(1,3)
[illegible]
    [illegible]
    [illegible]
```

说明 [illegible]

14.4.2 Java 接口应用

在这一节中，我们[illegible]在MATLAB中编[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]，[illegible]步骤详细介绍。

例 14.12 在MATLAB中编写用户电话簿日志程序的Java接口程序。

step 1 打开M文件编辑器，输入[illegible]代码

```
function phonebook(varargin)
pbname = 'myphone';
%处理原始数据文件的名称和路径
[illegible]
    datadir = char(java.lang.System.getProperty('user.dir'));
else
    datadir = getenv('HOME');
```

```
end;
pbname = fullfile(datadir, pbname)
%如果不存在文件，创建该文件
if ~exist(pbname)
   disp(sprintf('Data file %s does not exist.', pbname));
   r = input('Create a new phone book (y/n)?','s');
   if r == 'y',
      try
         FOS = java.io.FileOutputStream(pbname);
         FOS.close
      catch
         error(sprintf('Failed to create %s', pbname));
      end;
   else
      return;
   end;
end;
pb_htable = java.util.Properties;
try
   FIS = java.io.FileInputStream(pbname);
catch
   error(sprintf('Failed to open %s for reading.', pbname));
end;
pb_htable.load(FIS);
FIS.close;

while 1
%显示用户选择的选项
   disp ' '
   disp ' Phonebook Menu:'
   disp ' '
   disp ' 1. Look up a phone number'
   disp ' 2. Add an entry to the phone book'
   disp ' 3. Remove an entry from the phone book'
   disp ' 4. Change the contents of an entry in the phone book'
   disp ' 5. Display entire contents of the phone book'
   disp ' 6. Exit this program'
   disp ' '
%获取用户选择的选项
   s = input('Please type the number for a menu selection: ','s');
switch s
   case '1',
      name = input('Enter the name to look up: ','s');
      if isempty(name)
         disp 'No name entered'
      else
%调用查看函数
         pb_lookup(pb_htable, name);
      end;
  case '2',
%调用添加函数
      pb_add(pb_htable);
   case '3',
      name=input('Enter the name of the entry to remove: ', 's');
      if isempty(name)
         disp 'No name entered'
```

```
        else
%调用删除函数
            pb_remove(pb_htable, name);
        end;
    case '4',
        name=input('Enter the name of the entry to change: ', 's');
        if isempty(name)
            disp 'No name entered'
        else
%调用修改函数
            pb_change(pb_htable, name);
        end;
    case '5',
%调用显示列表函数
        pb_listall(pb_htable);
    case '6',
        try
            FOS = java.io.FileOutputStream(pbname);
        catch
            error(sprintf('Failed to open %s for writing.',...
                          pbname));
        end;
        pb_htable.save(FOS,'Data file for phonebook program');
        FOS.close;
        return;
    otherwise
        disp 'That selection is not on the menu.'
    end;
end;
```

上面的程序代码是该程序代码的主函数，在程序代码的开头首先处理电话号码文件的路径，如果用户在运行程序代码之前已经创建电话号码文件，则返回该文件的路径全称；如果没有创建电话号码文件，则重新创建java.io.FileOutputStream对象来添加电话号码数据。当程序代码创建电话号码对象后，则提供用户选择对应的操作，然后主函数将需要调用对应的子函数完成对应的操作。

step 2 添加所有的子函数程序代码。在M语言编辑器中输入下面的程序代码：

```
function pb_lookup(pb_htable,name)
entry = pb_htable.get(pb_keyfilter(name));
if isempty(entry),
   disp(sprintf('The name %s is not in the phone book',name));
else
   pb_display(entry);
end
%添加号码的子程序
function pb_add(pb_htable)
disp 'Type the name for the new entry, followed by Enter.'
disp 'Then, type the phone number(s), one per line.'
disp 'To complete the entry, type an extra Enter.'
name = input(':: ','s');
entry=[name '^'];
while 1
   line = input(':: ','s');
   if isempty(line)
```

```
        break;
    else
        entry=[entry line '^'];
    end;
end;
if strcmp(entry, '^')
    disp 'No name entered'
    return;
end;
%添加对应的电话号码
pb_htable.put(pb_keyfilter(name),entry);
disp ' '
disp(sprintf('%s has been added to the phone book.', name));
%删除号码的子程序
function pb_remove(pb_htable,name)
if ~pb_htable.containsKey(pb_keyfilter(name))
    disp(sprintf('The name %s is not in the phone book',name))
    return
end;
r = input(sprintf('Remove entry %s (y/n)? ',name), 's');
if r == 'y'
%删除选中的电话号码
    pb_htable.remove(pb_keyfilter(name));
    disp(sprintf('%s has been removed from the phone book',name))
else
    disp(sprintf('%s has not been removed',name))
end;
%修改号码的子程序
function pb_change(pb_htable,name)
entry = pb_htable.get(pb_keyfilter(name));
if isempty(entry)
    disp(sprintf('The name %s is not in the phone book', name));
    return;
else
    pb_display(entry);
    r = input('Replace phone numbers in this entry (y/n)? ','s');
    if r ~= 'y'
        return;
    end;
end;
disp 'Type in the new phone number(s), one per line.'
disp 'To complete the entry, type an extra Enter.'
disp(sprintf(':: %s', name));
entry=[name '^'];
while 1
    line = input(':: ','s');
    if isempty(line)
        break;
    else
        entry=[entry line '^'];
    end;
end;
%完成电话号码的修改
pb_htable.put(pb_keyfilter(name),entry);
disp ' '
disp(sprintf('The entry for %s has been changed', name));
```

```
%显示电话号码列表的子程序
function pb_listall(pb_htable)
enum = pb_htable.propertyNames;
while enum.hasMoreElements
   key = enum.nextElement;
%调用pb_display函数
   pb_display(pb_htable.get(key));
end;
%显示号码的子程序
function pb_display(entry)
disp ' '
disp '-------------------------'
[t,r] = strtok(entry,'^');
while ~isempty(t)
   disp(sprintf(' %s',t));
   [t,r] = strtok(r,'^');
end;
disp '-------------------------'
function out = pb_keyfilter(key)
if ~isempty(findstr(key,' '))
   out = strrep(key,' ','_');
else
   out = strrep(key,'_',' ');
end;
```

完成上面的程序代码后，将所有的程序代码保存为“phonebook.m”文件，然后将其保存到用户所使用的MATLAB路径中。

step 3 运行程序代码。在MATLAB的命令窗口中输入“phonebook”，得到如下的结果：

```
Phonebook Menu:

     1. Look up a phone number
     2. Add an entry to the phone book
     3. Remove an entry from the phone book
     4. Change the contents of an entry in the phone book
     5. Display entire contents of the phone book
     6. Exit this program

Please type the number for a menu selection: 5

-------------------------
 Sylvia Woodland
 (508) 111-3456
-------------------------
-------------------------
 Russell Reddy
 (617) 999-8765
-------------------------
```

step 4 添加新的数据。在上面的程序代码中，用户查看了原始的数据文件。在后面的步骤中，可以在该文件中添加新的数据，具体的信息如下：

```
Phonebook Menu:

     1. Look up a phone number
     2. Add an entry to the phone book
```

```
     3. Remove an entry from the phone book
     4. Change the contents of an entry in the phone book
     5. Display entire contents of the phone book
     6. Exit this program

Please type the number for a menu selection: 2

Type the name for the new entry, followed by Enter.
Then, type the phone number(s), one per line.
To complete the entry, type an extra Enter.
:: BriteLites Books
:: (781) 777-6868
::

BriteLites Books has been added to the phone book.

Phonebook Menu:

     1. Look up a phone number
     2. Add an entry to the phone book
     3. Remove an entry from the phone book
     4. Change the contents of an entry in the phone book
     5. Display entire contents of the phone book
     6. Exit this program

Please type the number for a menu selection: 5

---------------------------
 BriteLites Books
 (781) 777-6868
---------------------------

---------------------------
 Sylvia Woodland
 (508) 111-3456
---------------------------

---------------------------
 Russell Reddy
 (617) 999-8765
---------------------------
```

说明 用户还可以检测该程序代码的其他功能，在本章中，限于篇幅，在这里就不一一检测具体功能了，请用户自行尝试。

14.5 小结

在本章中，主要向读者介绍了在MATLAB中如何使用C或者FORTRAN语言创建MEX文件和MAT文件，然后介绍了MATLAB的引擎技术和Java接口的内容。这些内容是MATLAB程序接口的重要内容，希望用户仔细分析。

◆ ◆ ◆

反侵权盗版声明